SERV Маркина

Создано системой Doxygen 1.9.4

1 Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

2 Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

3 Список файлов

3.1 Файлы

Полный список документированных файлов.

4 Классы

4.1 Класс base

Класс базы данных

#include <base.h>

Граф связей класса base:

Открытые члены

- base (journal &log, std::string loc)
  Конструктор инициализации
- void read ()
  Метод чтения базы данных
- std::map< std::string, std::string > get_personal_data ()
  Геттер персональных данных
- ~base ()
  Деструктор модуля

Открытые атрибуты

- std::string base_loc
  Расположение файла базы данных
- std::ifstream database
  Объект потока ifstream для чтения базы данных
- std::map< std::string, std::string > data
  Словарь с персональными данными. Ключ - логин, пароль - значение
- journal & l
  Объект класса journal для записи сообщений в журнал

4.1.1 Подробное описание

Класс базы данных

Осуществляется работа с базой данных. Получение словаря с ключами - логинами, значеняими - паролями

4.1.2 Конструктор(ы)

4.1.2.1 base() base::base (journal & log, std::string loc)

Конструктор инициализации

Открывается файл базы данных,затем считывается содержимое,которое распределяется по словарю

Аргументы

in	log	Объект класса journal, передаваемый по ссылке
in	loc	Расположения файла базы данных

Исключения

critical_error,если	не удалось открыть файл с базой данных

4.1.2.2 ~base() base::~base ( )

Деструктор модуля

Закрывается файл с базой данных

4.1.3 Методы

4.1.3.1 get_personal_data() std::map< std::string, std::string > base::get_personal_data ( )

Геттер персональных данных

Возвращает

Словарь с персональными данными: логины, пароли

4.1.3.2 read() void base::read ( )

Метод чтения базы данных

Считывается содержимое, предстваленное в формате login/password

Исключения

critical_error,если	содержимое базы данных не соответствует заданному в коде формату

Объявления и описания членов классов находятся в файлах:

- base.h
- base.cpp

4.2 Класс communicator

Класс коммуникатора

#include <communicator.h>

Граф связей класса communicator:

Открытые члены

- void connect_to_client ()
  Соединение с клиентом
- int authentification ()
  Аутентификация клиента
- std::string SALT_generate ()
  Методи генерации соли
- std::string convert_to_hex (uint64_t)
  Конвертация числа в 16-ричную сс
- void send_data (const std::string &data, const std::string &msg)
  Отправка сообщения клиенту
- std::string recv_data (std::string messg)
  Прием данных от клиента
- std::string hash_gen (std::string &salt, std::string &password)
  Генерация хеша
- void close_sock ()
  Метод закрытия соединения с клиентом Соответствующе сообщение записывается в журнал
- void start ()
  Запуск сервера
- void work ()
  Метод начала работы сервера
- communicator (journal &log, base &base_r, uint p)
  Конструктор инициализации

Открытые атрибуты

- journal & l
  Объект класса journal для ведения записей
- base & b
  Объект класса base для получения персональных данных
- std::string id
  id, присланное клиентом
- int serverSocket
  Переменная для сокетов сервера и клиента
- int clientSocket

Закрытые данные

- struct sockaddr_in serverAddr clientAddr

  Структуры адреса сервера и клиента
- socklen_t addr_size

  Размер адреса
- std::map< std::string, std::string > data

  Словарь персональных данных
- size_t buflen =1024

  Размер буффера
- std::unique_ptr< char[]> buffer {new char[buflen]}

  Unique_ptr для приема и отправки сообщений
- uint port

  Порт

4.2.1 Подробное описание

Класс коммуникатора

Порт передается с помощью метода класса interface,работу сервера запускает метод work
Необходимые для работы объекты journal и base передаются по ссылке.

4.2.2 Конструктор(ы)

4.2.2.1 communicator() communicator::communicator (

```
journal & log,
base & base_r,
uint p )
```

Конструктор инициализации

Инициализация объектов journal, base по ссылке, установление порта

4.2.3 Методы

4.2.3.1 authentification() int communicator::authentification ( )

Аутентификация клиента

Сравнение присланного хеша со сгенерированным на основе пароля из базы данных При ошибке аутентификации закрывается соединение с клиентом

Возвращает

Функция возвращает 1, если не удалось аутентифицировать клиента, 0, если аутентификация успешна

4.2.3.2 connect_to_client() void communicator::connect_to_client ( )

Соединение с клиентом

Сервер ожидает подключение клиента

Возвращает

Функция ничего не возвращает

Исключения

fatal_error,если	не удалось встать в режим прослушки

4.2.3.3 convert_to_hex() std::string communicator::convert_to_hex (uint64_t x)

Конвертация числа в 16-ричную сс

Число с помощью потока stringstream конвертируется в 16-ричную сс

Аргументы

in	d_salt	Сгенерированное число

Возвращает

64-разрядное число

4.2.3.4 hash_gen() std::string communicator::hash_gen (std::string & salt, std::string & password)

Генерация хеша

Хеш генерируется на основе соли и пароля

Аргументы

in	salt	Соль
in	pswd	Пароль клиента

Возвращает

Функция возвращает сгенерированный хеш

4.2.3.5 recv_data() std::string communicator::recv_data (std::string messg)

Прием данных от клиента

В цикле принимается сообщение, если оно превышает размер буфера, то продолжается прием оставшейся части

Аргументы

in	msg	Строка для хранения принятого сообщения

Исключения

При	ошибке закрывается соединение с клиентом

4.2.3.6 SALT_generate() std::string communicator::SALT_generate ( )

Методи генерации соли

64 разрядное число конвертируется в 16-ричное с дополнением 0 до длины 16

Возвращает

Соль

4.2.3.7 send_data() void communicator::send_data (const std::string & data, const std::string & msg)

Отправка сообщения клиенту

Отправляется сообщение клиенту, конвертируя string в cstr

Аргументы

in	data	Строка для отправки
in	log_msg	Сообщение для журнала

Исключения

При	ошибке закрывается соединение с клиетом

4.2.3.8 start() void communicator::start ( )

Запуск сервера

Создание сокета и привязка к локальному адресу, запись персональных данных в словарь с помощью объекта base

Исключения

critical_error,если	не удалось создать или привязать сокет

4.2.3.9 work() void communicator::work ( )

Метод начала работы сервера

Запуск сервера, прием соединения от клиента, аутентификация, обработка присланных данных

Аргументы

in	log	Объект класса journal для записи сообщений в журнал
in	base↩ _r	Объект класса base для получения персональных данных из базы

Исключения

fatal_error	или закрытие соединения с клиентом в зависимости от характера ошибки

Объявления и описания членов классов находятся в файлах:

- communicator.h
- communicator.cpp

4.3 Класс data_handler

Класс обработки данных

#include <data_handler.h>

Граф связей класса data_handler:

Открытые члены

- int64_t calculator (int64_t number1, int64_t number2)
  Метод вычисления произведения вектора
- int handle_calculation ()
  Обработка данных от клиента
- data_handler (communicator &serv)
  Конструктор инициализации

Открытые атрибуты

- communicator & s

  Объект класса communicator для отправки результата и приема значений вектора
- uint32_t num_of_vec

  Количество векторов

4.3.1 Подробное описание

Класс обработки данных

Прием векторов от клиента и выполнение соответствующих вычислений, отправка результата вычислений

4.3.2 Конструктор(ы)

4.3.2.1 data_handler() data_handler::data_handler (communicator & serv) [inline]

Конструктор инициализации

Инициализация объектов communicator по ссылке

4.3.3 Методы

4.3.3.1 calculator() int64_t data_handler::calculator (int64_t number1, int64_t number2)

Метод вычисления произведения вектора

Аргументы

in	number1	Первый множитель
in	number2	Второй множитель

Предупреждения

При переполнении возвращается максимум или минимум типа int64_t

Возвращает

Произведение двух чисел

4.3.3.2 handle_calculation() int data_handler::handle_calculation ( )

Обработка данных от клиента

Прием значений вектора, выполнение соответствующих вычислений

Предупреждения

Если клиент отправил неверный тип данных, то соединение закрывается

Объявления и описания членов классов находятся в файлах:

- data_handler.h
- data_handler.cpp

4.4 Класс fatal_error

Класс ошибок

#include <error.h>

Граф наследования:fatal_error:

Граф связей класса fatal_error:

Открытые члены

- fatal_error (const std::string &s)
  Конструктор ошибки

4.4.1 Подробное описание

Класс ошибок

Обозначает специфические ошибки сервера

4.4.2 Конструктор(ы)

4.4.2.1 fatal_error() fatal_error::fatal_error (const std::string & s) [inline]

Конструктор ошибки

Аргументы

in	s	Сообщение об ошибке

Объявления и описания членов класса находятся в файле:

- error.h

4.5 Класс interface

Класс пользовательского интерфейса

#include <interface.h>

Открытые члены

- interface (int argc, char *argv[])
  Конструктор инициализации
- uint get_port ()
  Геттер порта
- std::string get_base_destination ()
  Геттер расоложения базы данных
- std::string get_log_destination ()
  Геттер расоложения файла журнала

Открытые атрибуты

- po::options_description desc
  Описание запрашиваемых параметров в комстроке
- po::variables_map vm
  Разобранные значения параметров комстроки

4.5.1 Подробное описание

Класс пользовательского интерфейса

Комстрока разбирается в конструкторе, обработана ситуация, когда не введен ни один из параметров Поддерживается ввод нескольких значений в параметр. Считается последнее введенное

4.5.2 Конструктор(ы)

4.5.2.1 interface() interface::interface (
int argc,
char * argv[])

Конструктор инициализации

Разбор комстроки

Аргументы

in	argc	Количество аргументов комстроки
in	argv	Значения аргументов комстроки

4.5.3 Методы

4.5.3.1 get_port() uint interface::get_port ( )

Геттер порта

Возвращает

Значение порта

Исключения

fatal_error	при системном порте

Объявления и описания членов классов находятся в файлах:

- interface.h
- interface.cpp

4.6 Класс journal

Класс записи сообщений в журнал

#include <journal.h>

Открытые члены

- journal (std::string destination)
  Конструктор инициализации
- int write_journal (std::string message)
  Метод записи сообщения в лог файл
- ~journal ()
  Деструктор

Открытые атрибуты

- std::ofstream l
  Объект ofstream для открытия файла журнала

4.6.1 Подробное описание

Класс записи сообщений в журнал

Обработана ситуация при закрытом файле после открытия, запись сообщений в методе write_↩ journal При ошибке открытия файла выключается сервер

4.6.2 Конструктор(ы)

4.6.2.1 journal() journal::journal (std::string destination)

Конструктор инициализации

Аргументы

in	destination	Расположение файла журнала

Открывается файл

Исключения

fatal_error,если	файла не существует

4.6.2.2 ~journal() journal::~journal ( )

Деструктор

Закрывается файл

4.6.3 Методы

4.6.3.1 write_journal() int journal::write_journal (std::string message)

Метод записи сообщения в лог файл

Записывается сообщение в формате время ===> сообщение Файл закрывается в конце работы

Аргументы

in	message	Сообщение для записи

Исключения

fatal_error,если	файл закрыт

Объявления и описания членов классов находятся в файлах:

- journal.h
- journal.cpp

5 Файлы

5.1 Файл base.cpp

Исполняемый файл модуля base.

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

5.2 Файл base.h

Заголовочный файл модуля base.

#include <vector>
#include <string>
#include <fstream>
#include <iostream>
#include <map>
#include <algorithm>
#include "journal.h"
#include "error.h"
#include <boost/filesystem.hpp>
Граф включаемых заголовочных файлов для base.h: Граф файлов, в которые включается этот файл:

Классы

- class base

  Класс базы данных

5.2.1 Подробное описание

Заголовочный файл модуля base.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.3 base.h

См. документацию.

```
1
8 #pragma once
9 #include <vector>
10 #include <string>
11 #include <fstream>
12 #include <iostream>
13 #include <map>
14 #include <algorithm>
15 #include "journal.h"
16 #include "error.h"
17 #include <boost/filesystem.hpp>
21 class base{
22     public:
26     std::string base_loc;
30     std::ifstream database;
34     std::map<std::string,std::string> data;
38     journal& l;
46     base(journal& log,std::string loc);
52     void read();
57     std::map<std::string,std::string> get_personal_data();
62     ~base();
63 };
```

5.4 Файл communicator.cpp

Исполняемый файл модуля communicator.

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

5.4.1 Подробное описание

Исполняемый файл модуля communicator.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.5 Файл communicator.h

Заголовочный файл модуля communicator.

```
#include <iostream>
#include <string>
#include <vector>
#include <cstring>
#include <algorithm>
#include <memory>
#include <stdlib.h>
#include <unistd.h>
#include <arpa/inet.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <sstream>
#include "data_handler.h"
#include <random>
#include <chrono>
#include <thread>
#include <limits>
#include "base.h"
#include "journal.h"
#include "error.h"
#include "interface.h"
#include <cryptopp/cryptlib.h>
#include <cryptopp/hex.h>
#include <cryptopp/files.h>
#include <cryptopp/md5.h>
#include <cryptopp/filters.h>
#include <cryptopp/osrng.h>
```

Граф включаемых заголовочных файлов для communicator.h: Граф файлов, в которые включается этот файл:

Классы

- class communicator

  Класс коммуникатора

Макросы

- #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1

5.5.1 Подробное описание

Заголовочный файл модуля communicator.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.6 communicator.h

См. документацию.

```
1
8 #pragma once
9 #include <iostream>
10 #include <string>
11 #include <vector>
12 #include <cstring>
13 #include <algorithm>
14 #include <memory>
15 #include <stdlib.h>
16 #include <unistd.h>
17 #include <arpa/inet.h>
18 #include <sys/socket.h>
19 #include <netdb.h>
20 #include <netinet/in.h>
21 #include <sstream>
22 #include "data_handler.h"
23 #include <random>
24 #include <chrono>
25 #include <thread>
26 #include <limits>
27 #include "base.h"
28 #include "journal.h"
29 #include "error.h"
30 #include "interface.h"
31 #include <cryptopp/cryptlib.h>
32 #include <cryptopp/hex.h>
33 #include <cryptopp/files.h>
34 #define CRYPTOPP_ENABLE_NAMESPACE_WEAK 1
35 #include <cryptopp/md5.h>
```

```
36 #include <cryptopp/filters.h>
37 #include <cryptopp/osrng.h>
42 class communicator{
43     private:
47         struct sockaddr_in serverAddr, clientAddr;
51         socklen_t addr_size;
55         std::map<std::string,std::string> data;
59         size_t buflen =1024;
63         std::unique_ptr <char[]> buffer{new char[buflen]};
67         uint port;
68     public:
72         journal& l;
76         base &b;
80         std::string id;
84         int serverSocket, clientSocket;
91         void connect_to_client();
98         int authentification();
104         std::string SALT_generate();
111         std::string convert_to_hex(uint64_t);
119         void send_data(const std::string& data, const std::string& msg);
126         std::string recv_data(std::string messg);
134         std::string hash_gen(std::string &salt,std::string &password);
139         void close_sock();
145         void start();
153         void work();
158         communicator(journal& log,base& base_r,uint p);
159
160 };
```

5.7 Файл data_handler.cpp

Исполняемый файл модуля data_handler.

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

5.7.1 Подробное описание

Исполняемый файл модуля data_handler.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.8 Файл data_handler.h

Заголовочный файл модуля data_handler.

#include "communicator.h"
#include "error.h"
#include "journal.h"
#include <limits>
#include <iostream>
#include <chrono>
#include <thread>
Граф включаемых заголовочных файлов для data_handler.h: Граф файлов, в которые включается этот файл:

Классы

- class data_handler

  Класс обработки данных

5.8.1 Подробное описание

Заголовочный файл модуля data_handler.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.9 data_handler.h

См. документацию.

```
1
8 #pragma once
9 #include "communicator.h"
10 #include "error.h"
11 #include "journal.h"
12 #include <limits>
13 #include <iostream>
14 #include <chrono>
15 #include <thread>
16 class communicator;
20 class data_handler{
21     public:
25     communicator& s;
29     uint32_t num_of_vec;
37     int64_t calculator(int64_t number1, int64_t number2);
43     int handle_calculation();
48     data_handler(communicator& serv):s(serv){}
49 };
```

5.10 Файл error.h

Заголовочный файл для модуля error.

#include <stdexcept>
#include <string>
Граф включаемых заголовочных файлов для error.h: Граф файлов, в которые включается этот файл:

Классы

- class fatal_error
  Класс ошибок

5.10.1 Подробное описание

Заголовочный файл для модуля error.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.11 error.h

См. документацию.

```
1
8 #pragma once
9 #include <stdexcept>
10 #include <string>
14 class fatal_error:public std::runtime_error{
15     public:
19     fatal_error(const std::string& s):std::runtime_error(s){}
20 };
```

5.12 Файл interface.cpp

Исполняемый файл модуля interface.

#include "interface.h"
#include "error.h"
#include <boost/program_options.hpp>
Граф включаемых заголовочных файлов для interface.cpp:

5.12.1 Подробное описание

Исполняемый файл модуля interface.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.13 Файл interface.h

Заголовочный файл модуля interface.

#include <boost/program_options.hpp>
#include <iostream>
#include <string>
#include <vector>
#include "error.h"
Граф включаемых заголовочных файлов для interface.h: Граф файлов, в которые включается этот файл:

Классы

- class interface

  Класс пользовательского интерфейса

5.13.1 Подробное описание

Заголовочный файл модуля interface.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.14 interface.h

См. документацию.

```
1
8 #pragma once
9 #include <boost/program_options.hpp>
10 #include <iostream>
11 #include <string>
12 #include <vector>
13
14 #include "error.h"
15 namespace po = boost::program_options;
20 class interface{
21     public:
24     po::options_description desc;
27     po::variables_map vm;
34     interface(int argc, char* argv[]);
40     uint get_port();
44     std::string get_base_destination();
48     std::string get_log_destination();
49 };
```

5.15 Файл journal.cpp

Исполняемый файл модуля journal.

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

5.15.1 Подробное описание

Исполняемый файл модуля journal.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.16 Файл journal.h

Заголовочный файл модуля journal.

#include <vector>
#include <string>
#include <fstream>
#include <iostream>
#include <chrono>
#include <cstring>
#include "error.h"
#include <boost/filesystem.hpp>
Граф включаемых заголовочных файлов для journal.h: Граф файлов, в которые включается этот файл:

Классы

- class journal

  Класс записи сообщений в журнал

5.16.1 Подробное описание

Заголовочный файл модуля journal.

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

5.17 journal.h

См. документацию.

```
1
8 #pragma once
9 #include <vector>
10 #include <string>
11 #include <fstream>
12 #include <iostream>
13 #include <chrono>
14 #include <cstring>
15 #include "error.h"
16 #include <boost/filesystem.hpp>
21 class journal{
22     public:
25         std::ofstream l;
32         journal(std::string destination);
40         int write_journal(std::string message);
45         ~journal();
46 };
```

5.18 Файл main.cpp

Главный файл проекта

#include "journal.h"
#include "base.h"
#include "communicator.h"
#include "interface.h"
#include "error.h"
#include "data_handler.h"
Граф включаемых заголовочных файлов для main.cpp:

Функции

- int main (int argc, char *argv[])

5.18.1 Подробное описание

Главный файл проекта

Автор

Маркина К.А.

Версия

1.0

Дата

21.06.2024

Авторство

ИБСТ ПГУ

Предметный указатель