Universidade de Coimbra
Faculdade de Ciências e Tecnologia de Coimbra
Departamento de Engenharia Informática

Introdução às Redes de Comunicação

Relatório

Trabalho 2 – Serviço de correio eletrónico simplificado

João Moreira 2015230374
Ricardo Tavares 2014230130

Introdução

Neste trabalho foram criados dois programas (um cliente e um servidor).

O programa cliente é utilizado como uma interface para controlar o que queremos fazer no servidor (seja enviar, receber ou ver emails). Para isto temos um menu de login em que inserindo um username e uma password temos acesso a todas as funcionalidades regulares. Temos também a opção de um utilizador se tornar num operador, passando a ter acesso a todo o conteúdo existente, .como alterar contas ou ver informação dos outros utilizadores.

O servidor tem a função de verificar tudo o que o cliente pretende fazer, sendo que verifica o login, verifica se o utilizador é um operador ou um utilizador normal, e é ele que envia as mensagens entre utilizadores. Guarda também as mensagens em memória, assim como alterações que utilizadores possam vir a fazer.

Estes programas comunicam através de sockets.

Manual de Utilizador

Para executar os programas é necessário compilar os ficheiros client.c e server.c:

gcc -pthread server.c –o server

gcc client.c -o client

De seguida, correm-se os executáveis em duas janelas de terminal diferentes.

Servidor: ./server

Querendo definir a porta do servidor manualmente, correr passando -p e a porta como argumento: ./server <-p> <porta>

Cliente: ./client <host> <porta>

O menu do programa é composto por várias opções, cada uma identificada com um número para ser executada, assim como uma breve descrição do que faz.

Algumas funções pedem atributos de operador, que é obtido selecionando a opção nº 7 e introduzindo as credenciais

username “su” password “su”

Descrição das classes

header.h

Este ficheiro contem todas as funções que são utilizadas ao longo do programa, tanto as do servidor como do cliente.

Bibliotecas:

```
#include <stdio.h>
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <unistd.h>
#include <stdlib.h>
#include <netdb.h>
#include <string.h>
#include <signal.h>
#include <pthread.h>
```

Variáveis Globais:

```
#define BUF_SIZE 1024
#define LOGINSIZE 9
#define SERVER_PORT 9000
#define MSGSIZE 250
#define STRSIZE 30
#define THREADS 5
```

Estruturas de armazenamento de dados :

```
typedef struct request_server {
        char username[LOGINSIZE];
        char password[20];
} request_server;
```

Esta estrutura é utilizada para enviar ao servidor os dados do utilizador que pretende aceder ao seu email.

```
struct email {
  int read;
  int nemail;
  char sender[LOGINSIZE];
  char reciever[LOGINSIZE];
  char assunto[STRSIZE];
  char msg[MSGSIZE];
  struct email* next;
  struct email* prev;
};
```

A partir desta estrutura obtemos as características necessárias ao servidor para que possamos caracterizar o utilizador e a mensagem que ele quer enviar, ou seja, remetente, destinatário, assunto e a mensagem em si.

```
struct userNode {
  char username[LOGINSIZE];
  char password[20];
  int numemails;
  int lastnum;
  int grupo;
  int oper;
  struct email* listaEmails;
  struct userNode* next;
}; typedef struct userNode Users;
```

Com esta estrutura temos todo o tipo de variáveis que assumimos que seriam indispensáveis à execução das funções pedidas para o servidor de Email.

server.c

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

Na função main criamos o sinal que controla a saída do server (ctrl + c). Para além disso utilizamos a função serverStart() que irá iniciar o servidor em si.

- **void erro (char *msg)**

É nesta função que se imprime no ecrã o erro que o servidor obteve, caso obtenha algum erro.

- **void end (int signum)**

Esta é a função de controlo do ctrl + c, sendo que ao premir estas teclas iremos desligar o servidor fechando a socket.

- **void serverStart()**

Aqui criamos as sockets com as quais vamos fazer a comunicação com o cliente e as threads que ajudam no controlo do servidor.

- **void worker(int id)**

A função worker é o inicio do ciclo de leitura dos pedidos do cliente, sendo que nele temos um ciclo while que faz a leitura caso algum cliente tente aceder ao servidor. Ele faz a confirmação do login.

- **Users* login (char* username, char* password)**

Função que verifica se o username e a password coincidem com os existentes.

- **void load_users (char *txt)**

Leitura do ficheiro txt das informações dos usuários.

- **void create_user(char *usertmp, char *passtmp)**

Alocação de memoria nos quais inserimos os usuários lidos do ficheiro txt.

- **void server_menu(Users* node, int client_fd)**

Menu onde a opcao escolhida vai ao encontro da funcao pretendida pelo cliente.

Nesta função englobamos:

void listmess();
void listusers();
void sendmess();
void listread();
void removemes();
void changepass();
void oper();
void openmsg();
void createacc();
void deluser();

client.c

- **init_client(int argc, char **argv)**

Aqui criamos as sockets com as quais vamos fazer a comunicação com o server.

- **void start()**

Esta função recebe os dados do utilizador e envia-os ao servidor para ele confirmar se eles estão inseridos no servidor, caso estejam o cliente tem acesso a à função menu().

- **menu()**

Com esta função o cliente tem uma variedade de funcionalidades para escolher, sendo que estas são:

Listar todas as mensagens por ler - **void listmess()**;
Listar todos os clientes autorizados - **void listusers()**;
Enviar uma mensagem para um cliente autorizado - **void sendmess()**;
Listar todas as mensagens já lidas - **void listread()**;
Apagar mensagens - **void removemes()**;
Mudar a password - **void changepass()**;
Obter os privilégios de operador - **void oper()**;
Sair do Email – não tem uma função porque apenas fazemos um break tanto no cliente como no servidor.
Abrir mensagens – **void openmsg()**;
Criar conta (apenas para operadores) – **void createacc()**;
Eliminar utilizadores (apenas para operadores) – **void deluser()**; (não concluída)