1 Servidor DS - Organização de dados

O servidor DS instalado na máquina 'tejo' do laboratório LT5 usa uma estrutura de directorias para armazenar toda a informação referente ao protocolo de forma persistente entre sessões.
Os alunos podem replicar a estrutura de directorias aqui descrita, que foi concebida tendo em vista simplificação do processamento quer para armazenamento flexível e indexação de utilizadores, grupos e mensagens, aproveitando assim as estruturas de dados previstas no *filesystem* do Linux.

Na directoria de trabalho do servidor DS encontram-se duas directorias:

USERS
e
GROUPS

Assume-se que no arranque do servidor DS estas duas directorias já estão criadas. Todas as restantes directorias e ficheiros descritos neste guia são criados ou eliminados pelo servidor de acordo com as operações previstas no protocolo.

1.1 A directoria USERS

A directoria USERS contém informação sobre os utilizadores registados e os utilizadores com sessões activas. Sob a directoria USERS vai existir uma directoria por cada utilizador registado. Essa directoria de utilizador será criada aquando da operação de *register* bem sucedida para o respectivo utilizador.

USERS/uid1 - directoria do user com identificação uid1
/uid2 - directoria do user com identificação uid2

Cada directoria uid(n) (a designação uid(n) é composta sempre por 5 algarismos) é criada quando o servidor recebe uma mensagem REG válida para um utilizador se registar. Dentro de cada directoria uid(n) será criado também um ficheiro designado **uid(n)_pass.txt** contendo a password do utilizador registado. Este ficheiro é criado imediatamente a seguir à criação da directoria do utilizador.
Se o utilizador tiver uma sessão activa (iniciada com o comando *login*) existe também na directoria uid(n) um ficheiro designado **uid(n)_login.txt** criado quando o servidor recebe uma mensagem LOG válida. O ficheiro **uid(n)_login.txt** deve ser eliminado quando o utilizador efectua *logout*.

Quando o utilizador emite o comando *unregister*, a directoria uid(n) (em USERS) é eliminada juntamente com os ficheiros que possa conter nesse momento.

Exemplifica-se a seguir uma possível utilização da directoria USERS:

Quando o utilizador 10101 se regista com sucesso é criada a directoria '10101' sob USERS:

USERS/10101

De seguida, e sob a directoria recém criada, é imediatamente criado o ficheiro '10101_pass.txt' contendo a password que o utilizador escolheu:

USERS/10101/10101_pass.txt

A directoria '10101' e o ficheiro '10101_pass.txt' só serão eliminados se o respectivo utilizador fizer *unregister* com sucesso.

Se o utilizador 10101 fizer *login* com sucesso é criado sob USERS/10101 o ficheiro '10101_pass.txt' ficando agora dois ficheiros sob a directoria USERS/10101:

USERS/10101/10101_pass.txt
/10101_login.txt

Assim que o utilizador 10101 fizer *logout*, o ficheiro '10101_login.txt' é eliminado.

Salientam-se aqui as vantagem de uma estrutura de dados persistentes deste tipo:

1 - Quando o servidor precisa de acrescentar ou eliminar o registo de um UID, não tem de lidar com estruturas de tamanho variável por si geridas. Usa as capacidades que o sistema operativo já oferece em articulação com o *filesystem*, sem limitações de espaço no contexto deste projecto.

2 - Quando o servidor precisa de saber se um dado UID está registado ou se está em sessão, não precisa de efectuar buscas sobre estruturas de dados de tamanho variável por si geridas. Basta-lhe tentar abrir os ficheiros em causa para encontrar de imediato a informação desejada, ou perceber que ela não existe.

3 - Se o servidor terminar abruptamente ficam salvaguardados todos os seus dados, podendo alguma inconsistência menor ser resolvida manualmente por edição, criação ou eliminação de ficheiros e/ou directorias, com a possibilidade adicional de *debug* usando a data de

modificação de ficheiros disponibilizada pelo sistema operativo.

1.2 A directoria GROUPS

A directoria GROUPS contém toda a informação sobre os grupos existentes, os seus nomes, os utilizadores que lhes estão associados (com *subscribe*) e as mensagens publicadas (com *post*) e acessíveis aos utilizadores associados ao cada grupo.

Assim, a directoria GROUPS contém uma directoria gid(n) que será criada por cada novo grupo criado:

GROUPS/gid1 - directoria do grupo com identificação gid1
/gid2 - directoria do grupo com identificação gid2

A designação genérica gid(n) será formada por dois algarismos. De 01 a 99.
No momento de criação da directoria genérica gid(n) são criados de imediato sob a mesma:
- O ficheiro **gid(n)_name.txt** contendo o nome do grupo em causa.
- Uma sub-directoria designada MSG para conter as futuras directorias de mensagens à medida que estas forem sendo publicadas.

Para além disso, dentro da directoria genérica gid(n) serão criados sucessivamente ficheiros designados **uid(n).txt**. Um por cada utilizador que se associar ao grupo (através de *subscribe*). Sempre que um utilizador revogar a sua associação ao grupo, o respectivo ficheiro **uid(n).txt** será eliminado.

A estrutura de directorias para armazenamento das mensagens publicadas é exemplificada como segue:

GROUPS/gid1/MSG/mid1 - directoria da mensagem com identificação mid1, publicada para o grupo gid1.
/mid2 - directoria da mensagem com identificação mid2, publicada para o grupo gid1.

Cada directoria de mensagem mid(n) é criada quando um utilizador faz *post*.
Cada directoria mid(n) contém obrigatoriamente dois ficheiros:
- Um ficheiro designado **T E X T.txt** contendo o texto da mensagem.
- Um ficheiro designado **A U T H O R.txt** contendo o UID do autor da mensagem.

Cada directoria mid(n) pode conter ainda, opcionalmente, um ficheiro adicional que possa ter sido publicado com o *post* da mensagem respectiva.

Exemplifica-se a seguir uma lista de directorias sob a directoria GROUPS e seus conteúdos para o grupo com identificação '01':

GROUPS/01/01_name.txt
GROUPS/01/11111.txt
GROUPS/01/22222.txt
GROUPS/01/MSG/0001/A U T H O R.txt
GROUPS/01/MSG/0001/T E X T.txt
GROUPS/01/MSG/0002/A U T H O R.txt
GROUPS/01/MSG/0002/T E X T.txt
GROUPS/01/MSG/0002/photo.jpg

- O ficheiro **01_name.txt** contém o nome do grupo '01'.
- O ficheiro 11111.txt indica que o utilizador com identificação 11111 está associado ao grupo '01'.
- O ficheiro 22222.txt indica que o utilizador com identificação 22222 está associado ao grupo '01'.
- A directoria /MSG/0001 foi criada para armazenar a primeira mensagem do grupo '01'. Esta directoria contém o ficheiro com o texto da mensagem e o ficheiro com a identificação do seu autor.

- A directoria /MSG/0002 foi criada para armazenar a segunda mensagem do grupo '01'. Esta directoria contém o ficheiro com o texto da mensagem, o ficheiro com a identificação do seu autor e ainda um ficheiro designado **photo.jpg** carregado com a mensagem em causa aquando da emissão do comando *post* na aplicação user.

Os ficheiros **A U T H O R.txt** e **T E X T.txt** contêm propositadamente espaços nas suas designações para evitar que um ficheiro opcional carregado com o comando *post* se lhes possa sobrepor por coincidência de designação.

O modelo acima descrito para os grupos apresenta o mesmo tipo de benefícios que já foram referidos para a base de dados dos utilizadores. Acresce que por conterem os grupos dados mais diversos e volumosos (nomeadamente os ficheiros associados às mensagens) se revela particularmente vantajoso adoptar um tal modelo de organização de dados.

2 USERS e GROUPS especiais para testes

Com a finalidade de controlar e preservar exclusivamente para testes de consulta alguma da informação existente na base de dados do servidor DS, aos UID e aos GID começados por zero estão vedadas todas as operações que possam alterar os dados de utilizadores e de grupos, nomeadamente a inserção de novas mensagens por esses UID e para esses GID.

2.1 USERS especiais

Para os UID começados por zero pré-existentes na base de dados do servidor DS na máquina 'tejo', apenas são possíveis operações que não impliquem alterações na base de dados.
- *login* - é efectuada a verificação da password, mas não é mantido registo sobre a sessão.
- *logout* - é possível mas não produz efeito.
- *groups*
- *my_groups*

- *select* - comando de efeito apenas local
- *ulist*
- *retrieve*

Os primeiros nove utilizadores com UID entre 00001 e 00009 têm *passwords* respectivamente pword001 a pword009.
Os mesmos nove utilizadores estão inscritos em todos os grupos com GID entre 01 e 09, maior ou igual ao seu UID. Assim, o utilizador 00001 está inscrito em todos os grupos com GID de 01 a 09, enquanto o utilizador 00009 está inscrito apenas no grupo 09.

2.2 GROUPS especiais

Existem nove grupos especiais na base de dados do servidor DS numerados de 01 a 09 para os quais só são possíveis as seguintes operações:

- *groups*
- *my_groups*
- *select*
- *ulist*
- *retrieve*

Apresentam-se a seguir os nomes atribuídos e resumos de conteúdos dos grupos com GID de 01 a 09:

- Grupo 01: Two-messages-text-only
Este grupo contém duas mensagens com texto apenas. O texto da primeira mensagem contém um caracter. O texto da segunda mensagem contém 240 caracteres.

- Grupo 02: One-msg-text-and-photo

Este grupo contém apenas uma mensagem com um ficheiro jpg com cerca de 17MBytes.

- Grupo 03: Two-msg-Text-Photo

Este grupo contém duas mensagens. A primeira só tem texto. A segunda tem também o ficheiro jpg do grupo 02.

- Grupo 04: Supersonic-planes

Este grupo contém duas mensagens com dois ficheiros jpg cada uma.

- Grupo 05: Great-Battles

Este grupo contém 21 mensagens. Cada mensagem contém um ficheiro com imagem.

- Grupo 06: Great-Battles-Pics-ODD

Este grupo contém as mesmas mensagens do grupo 05. No entanto só as mensagens ímpares contêm ficheiros de imagem.

- Grupo 07: Great-Battles-Pics-EVEN

Este grupo contém as mesmas mensagens do grupo 05. No entanto só as mensagens pares contêm ficheiros de imagem.

- Grupo 08: Battle-of-Lake-Peipus

Este grupo contém duas mensagens, ambas com ficheiros. A primeira mensagem contém um ficheiro com cerca de 53MBytes. A segunda mensagem contém um ficheiro com cerca de 2MBytes.

- Grupo 09: 9

Este grupo não contém mensagens. O nome do grupo contém apenas um caracter.

A estrutura de dados para os grupos acima descritos encontra-se no ficheiro GROUPS.zip publicada em anexo a este guia. Os ficheiros contidos nas mensagens de GROUPS.zip, encontram-se no ficheiro MSG_FILES.zip. também publicado em anexo a este guia.

3 Complementos de programação para tratamento de directorias e ficheiros

Com a finalidade de auxiliar a concretização de uma estrutura de dados como aquela que se apresentou na secção 1, ilustram-se aqui algumas funções usadas pelo servidor DS para a criação, leitura e remoção de directorias e eliminação de ficheiros usando funções de interacção com

o *filesystem* definidas na linguagem C.

As funções aqui exemplificadas podem ser generalizadas para utilizações semelhantes no contexto do projecto.

3.1 Criação de directoria de utilizador após recepção de mensagem REG

```
#include <unistd.h>

int CreateUserDir(char *UID)
{
    char user_dirname[20];
    int ret;

    sprintf(user_dirname,"USERS/%s",UID);
    ret=mkdir(user_dirname,0700);
    if(ret==-1)
        return(0);
    return(1);
}
```

3.2 Eliminação de directoria de utilizador após recepção de mensagem UNR

```
#include <unistd.h>
```

```
int DelUserDir(char *UID)
{
    char user_dirname[20];
    sprintf(user_dirname,"USERS/%s",UID);



    if(rmdir(user_dirname)==0)
        return(1);
    else
        return(0);
}
```

3.3 Leitura do conteúdo de uma directoria

A função ilustrada abaixo lê o conteudo da directoria 'GROUPS' para preencher uma lista de todos os grupos existentes na base de dados. Para cada grupo, lê também o ficheiro existente na subdirectoria respectiva contendo o nome do grupo.
No final da execução, a estrutura cujo ponteiro se designa por 'list', fica preenchida com todos os GID de todos os grupos e os respectivos nomes.

```
#include <dirent.h>


int ListGroupsDir(GROUPLIST *list)
{
    DIR *d;
    struct dirent *dir;
    int i=0;
```

```
FILE *fp;
char GIDname[30];



list->no_groups=0;

d = opendir("GROUPS");

if (d)
{
    while ((dir = readdir(d)) != NULL)
    {
        if(dir->d_name[0]=='.')
            continue;
        if(strlen(dir->d_name)>2)
            continue;
        strcpy(list->group_no[i], dir->d_name);
        sprintf(GIDname,"GROUPS/%s/%s_name.txt",dir->d_name,dir->d_name);
        fp=fopen(GIDname,"r");
        if(fp)
        {
            fscanf(fp,"%24s",list->group_name[i]);
            fclose(fp);
        }
        ++i;
```

```
            if(i==99)
                break;
        }
        list->no_groups=i;
        closedir(d);
    }
    else
        return(-1);

    if(list->no_groups>1)
        SortGList(list);

    return(list->no_groups);
}
```

3.4 Eliminação de ficheiro de password após recepção de mensagem UNR

A função DelPassFile é invocada pelo servidor DS no contexto de uma operação *unregister* para eliminar o ficheiro que contém a password do utilizador. Esta função é chamada antes da função DelUserDir().

```
#include <unistd.h>

int DelPassFile(char *UID)
{
    char pathname[50];
    sprintf(pathname,"USERS/%s/%s_pass.txt",UID,UID);
```

```
    if(unlink(pathname)==0)
        return(1);
    else
        return(0);
}
```

4 Temporização de sockets

As duas funções que se seguem exemplificam a programação de *timers* em sockets para leitura:

```
#include <sys/types.h>
#include <sys/socket.h>



int TimerON(int sd)
{

    struct timeval tmout;

    memset((char *)&tmout,0,sizeof(tmout)); /* clear time structure */
    tmout.tv_sec=15; /* Wait for 15 sec for a reply from server. */
    return(setsockopt(sd, SOL_SOCKET, SO_RCVTIMEO,
                            (struct timeval *)&tmout,sizeof(struct timeval)));
}
```

```
int TimerOFF(int sd)
{

    struct timeval tmout;

    memset((char *)&tmout,0,sizeof(tmout)); /* clear time structure */
    return(setsockopt(sd, SOL_SOCKET, SO_RCVTIMEO,
                           (struct timeval *)&tmout,sizeof(struct timeval)));
}
```