Introdução às Redes e Comunicação

Ano Lectivo de 2015/2016

Trabalho 2

Protocolo de transferência de ficheiros com cache

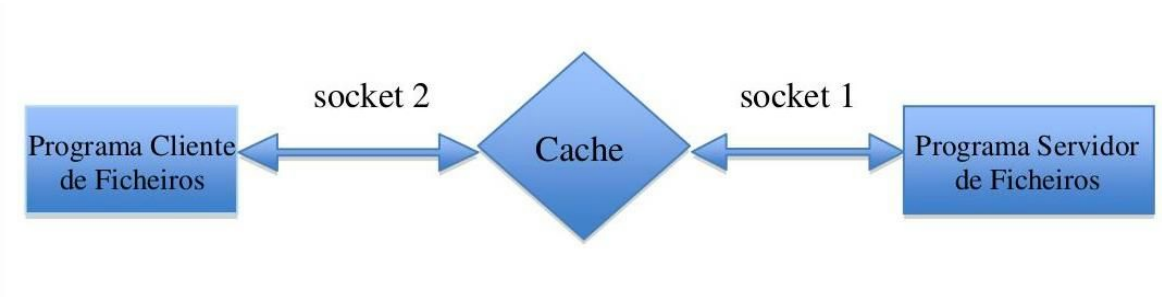
Relatório de trabalho realizado por:
António Simões, Nº 2014198322

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

Introdução

Linguagem Escolhida - **PYTHON 2.7.6**
Apesar de várias fichas práticas que envolveram comunicação entre sockets terem sido feitas em C, decidi fazer este trabalho em Python por uma maior simplicidade e estabilidade em várias etapas da comunicação comparadamente a C. A escolha da versão 2.7 em detrimento de uma mais recente prende-se com o facto de me ser mais familiar.

A comunicação entre o Cliente e os dois Servidores, Cache e Servidor de ficheiros, é feita através de dois sockets: um socket entre o cliente e a cache, e outro socket entre a Cache e o Servidor.



Para os alunos de Engenharia Informática, que é o meu caso, é necessária a implementação de um sistema de autenticação por login e password, de forma a que apenas utilizadores possam utilizar o sistema. Este sistema de autenticação não envolve o servidor cache portanto, nesta situação, é feita a ligação Cliente-Servidor diretamente através de um só socket.

Posto isto e dividindo em 2 situações:

1) - Ligação Cliente - Cache - Servidor (Sem Login)
2) - Ligação Cliente - Servidor (Com Login)

Ligação Cliente - Cache - Servidor

Nesta situação, o cliente apenas se conecta à Cache e esta conecta-se ao servidor, atuando portanto como cliente para o servidor de ficheiros e como servidor para o cliente.
O servidor cache, de modo a garantir ligações simultâneas, após a criação do socket TCP, vai estar à escuta de varias ligações. Quando algum cliente se conecta, um Thread vai ser criado de modo a tratar das várias operações enquanto que o processo principal continua à espera de novas ligações.

```
while True:
      print 'Cache server listening at port', cachePort, '...'
      connection, addr = sock.accept()
      thread.start_new_thread(operations, tuple([connection,addr]))
```

Dentro da função **operations()** o servidor vai estar à espera para receber mensagens do cliente com a seguinte estrutura:
Mensagem = “operação**\n**username**\n**password”
(‘**\n**’ é usado para facilitar o processamento dos vários parâmetros a receber)
Ao ser recebida a mensagem, processa-se a opção e segundo o critério apresentado de seguida, passa a realizar as funções pedidas:

Operações	Código recebido pelo Servidor
LIST	3
UPLOAD	4
DOWNLOAD	5
QUIT	6

```
while option != 6:
      if option == 3:    # LISTAR
            listar(sock_sv, conn)
      elif option == 4: # DOWNLOAD
            download(sock_sv, conn)
      elif option == 5: # UPLOAD
            upload(sock_sv, conn)
```

Ligação Cliente - Servidor (Com Login)

O sistema de login funciona da mesma maneira que as outras operações acima referidas, com a excepção de que se o login ou a criação de um novo utilizador falharem, a ligação é quebrada. Quando o login é bem sucedido, a ligação mantém-se aberta até o servidor receber a opção de número 6 para o propósito de fecho do cliente (ou simplesmente logout).

Para armazenar a combinação de utilizadores/passwords é usado um dicionário, este que é guardado num ficheiro de objetos de formato ‘pkl’ que é escrito e carregado pelo servidor sempre que executa.

O formato ‘pkl’ provém do módulo pickle (ou cPickle) em python, que auxilia no processamento de ficheiro com objetos tais como: listas ou dicionários, no caso deste programa, entre outros.

Para escrever no ficheiro: **pickle.dump(...)**
Para ler do ficheiro, no arranque: **pickle.load(...)**

O dicionário com os utilizadores e passwords é guardado desta forma assim como a lista de ficheiros que cada utilizador tem armazenado no seu respetivo diretório. Cada diretório de utilizador terá um ficheiro pickle que armazena uma lista de ficheiros presentes na pasta, após o cliente ter feito upload dos mesmos. Este ficheiro ‘**list.pkl**’ é carregado para uma lista dentro do programa - ‘**file_list**’.

Na situação anterior que passa pela cache, visto que o servidor está construído para tratar de ambas as situações, é enviado o utilizador ‘default’ quando se está a passar pelo servidor que faz de bridge.

Operações	Código recebido pelo Servidor
LOGIN	1
NOVO UTILIZADOR	2