**Trabalho 2**

Configuração de uma Rede e Desenvolvimento de uma Aplicação de Download

**Relatório Final**



Mestrado Integrado em Engenharia Informática e Computação

**Redes de Computadores**

**Grupo 4 :**

Ana Rita Torres, [up201406093@fe.up.pt](mailto:up201406093@fe.up.pt)

Catarina Correia,  [up201405765@fe.up.pt](mailto:up201405765@fe.up.pt)

Ricardo Neves, [up201405868@fe.up.pt](mailto:up201405868@fe.up.pt)

Faculdade de Engenharia da Universidade do Porto

R. Dr. Roberto Frias, 4200-464 Porto

20 de dezembro de 2016

# Sumário

Este relatório tem como objetivo explicar o segundo projeto da Unidade Curricular de Redes de Computadores. Este projeto dividiu-se em duas partes: numa primeira parte, foi desenvolvida uma aplicação de *download;* numa segunda parte, foram realizadas seis experiências especificadas no enunciado com o intuito de configurar uma rede.

Nas secções que se seguem, será explorado o desenvolvimento da referida aplicação e será feita uma análise das experiências mencionadas.

# Índice

# Introdução

O segundo projeto de Redes de Computadores, como já foi referido, divide-se um duas partes principais. Na primeira parte, foi pedida a elaboração de uma aplicação de *download* que procedesse à transferência de um ficheiro, implementando o protocolo *FTP*. Na segunda parte, foi pedida a configuração de uma rede. Este configuração foi dividida em várias experiências e respeita a seguinte ordem:

1. Configuração de um *IP* de rede;
2. Implementação de duas redes *LAN’s* virtuais num *switch*;
3. Configuração de um *router* em *LINUX*;
4. Configuração de um *router* comercial com e implementação de *NAT*;
5. *DNS;*
6. Conexões *TCP;*

# Parte 1 – Aplicação de Download

Para a realização desta primeira parte, o grupo baseou-se em vários documentos destacando-se os seguintes: o ficheiro *RFC959* que descreve o protocolo de transferência de dados (*FTP*) e o ficheiro *RFC1738* que informa acerca do uso de *URL* e o seu devido tratamento.

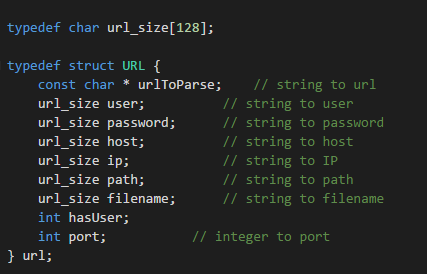
## Arquitetura

Para uma melhor organização e estrututa, a aplicação divide-se em duas camadas: a de processamento do *URL* e a do cliente *FTP*.

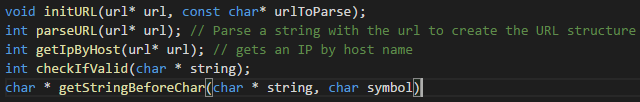
A aplicação desenvolvida aceita um *link* como argumento, especificado na linha de comandos e permite que o *download* seja feito de forma anónima ou não. Caso não se pretenda o anonimato, basta fornecer no *URL* o utilizador, seguido por ‘:’, a palavra-passe e por fim ‘@’. Se tais dados não forem fornecidos, assume-se o utilizador como anónimo (“*anonymous”*) e a palavra-passe como sendo nula.

/\* imagens de input com e sem utilizador \*/

Relativamente ao processamento do *URL,* foi criada uma *struct* que guarda as várias informações representadas no *link*: *urlToParse, user, password, host, ip, path, filename, hasUser* e *port*. O atributo *port* é sempre 21, pois é o número de controlo do protocolo *FTP.*



Na execução do programa, são chamadas várias funções:



A função ***initURL*** guarda o *URL* recebido e aloca memória para os vários atributos. Em seguida, a ***parseURL*** processa a variável *urlToParse* (*link* recebido na linha de comandos) *e* guarda toda a informação necessária. Por último, a função ***getIPByHost***, que chama a função *gethostbyname* com o *host* da *struct,* converte o *hostname* para um endereço *IP.*

As funções ***checkIfValid*** e ***getStringBeforeChar*** são funções auxiliares da *parseURL* e verificam se o *URL* recebido é constituído por caracteres válidos e obtêm uma *string* antes de um determinado caracter, respetivamente.

/\* Falar do ftp – RITA \*/

## Resultados de Download

A aplicação desenvolvida foi testada não só em modo normal, ou seja, com um utilizador e palavra-passe, mas também em modo anónimo. Para efeito de teste, foram realizados diversos *dowloads*, todos bem sucedidos, tendo o maior ficheiro testado 500MB.

Caso ocorra algum erro, a aplicação termina e o erro é impresso na consola. Caso contrário, é impresso na consola uma mensagem a dizer que a transferência foi bem sucedida e qual o tamanho do ficheiro.

# Parte 2 – Configuração da rede e análise

## Configuração de um IP de rede

Esta experiência teve como objetivo não só compreensão da configuração de IP’s em máquinas diferentes, numa mesma rede, de forma a que estas pudessem comunicar entre si, mas também a identificação e distinção dos diferentes pacotes enviados.

Existem vários tipos de pacotes de dados, como por exemplo os ARP. Este protocolo é responsável por mapear um endereço de rede para um endereço físico (MAC). Quando um pacote chega a um *gateway*, este pede ao ARP para encontrar um host físico ou um endereço MAC que corresponda ao endereço IP. Por sua vez, este procura na sua cache e, se nenhuma entrada for encontrada, o ARP transmite um pacote de solicitação, que contém o IP para o qual se pretende saber o MAC, para todas as máquinas da mesma LAN. Se, em alguma máquina, ocorrer uma correspondência, então esta envia um pacote que contém o seu endereço MAC à máquina que solicitou a informação e o ARP guarda na sua tabela os dados para uma próxima ocorrência.

Cada pacote de dados contém, no cabeçalho das tramas enviadas, um identificador constituído por 2 bytes, *EtherType*, que permite identificar o protocolo associado. Por exemplo, no caso do ARP, o identificador é 0x0806 e, no caso do IP (Ipv4), o identificador corresponde ao valor 0x0800.

O tamanho de cada pacote enviado é <http://serverfault.com/questions/87927/how-does-ethernet-know-how-long-a-frame-is>

<https://en.wikipedia.org/wiki/Ethernet_frame#Ethernet_II>

Para a realização desta experiência, primeiro foram configurados os IP’s das portas eth0 de dois computadores, o tux41 e o tux44, utilizando o comando ***ifconfig*** e o comando ***route***, necessário para adicionar rotas à tabela de reencaminhamento.

Em seguida, enviou-se o comando ***ping*** para verificar a conetividade entre as duas máquinas configuradas. Este comando gera e envia para o endereço escolhido pacotes ICMP (um protocolo que relata erros caso um determinado serviço ou host não possa ser alcançado para o envio de pacotes) e aguarda por uma resposta.

Deste modo, foi possível observar, através do *Wireshark*, os pedidos ARP e a resposta da máquina correspondente com o respetivo endereço MAC.

## Implementação de duas redes *LAN’s* virtuais num switch

## Configuração de um *router* em LINUX

## Configuração de um *router* comercial e implementação de *NAT*

## *DNS*

## Conexões *TCP*