Universidade de Coimbra
Faculdade de Ciências e Tecnologia

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

TRABALHO PRÁTICO

Ano Letivo de 2019/20

Objetivo geral

O objetivo deste trabalho é implementar um sistema cliente-servidor para transferência de ficheiros entre dois sistemas, recorrendo para o efeito aos protocolos de transporte TCP ou UDP. A aplicação deverá permitir igualmente encriptar os dados transferidos entre o servidor e o cliente, com recurso a uma biblioteca de criptografia. Por último, da aplicação fará parte um *proxy*, capaz de mediar as comunicações entre o cliente e o servidor, tal como descrito a seguir.

Tal como referido anteriormente, o sistema a desenvolver consiste em três componentes: o cliente, o servidor e o *proxy*. O proxy permitirá intercetar as comunicações, com o objetivo de validar o funcionamento da aplicação e simular diferentes condições de transmissão. Descrevem-se, a seguir, de forma mais detalhada as funcionalidades a implementar em cada uma das componentes do sistema a desenvolver.

Funcionamento geral da aplicação

O servidor deve permitir o acesso simultâneo a no máximo N clientes, e cada cliente poderá, através da rede, listar e fazer transferência dos ficheiros que se encontram no servidor (ficheiros com diferentes características, por exemplo de texto, música ou vídeo). Os ficheiros podem ser transferidos por UDP ou TCP, com ou sem encriptação dos dados. A figura seguinte ilustra o modelo pretendido para a aplicação a desenvolver.

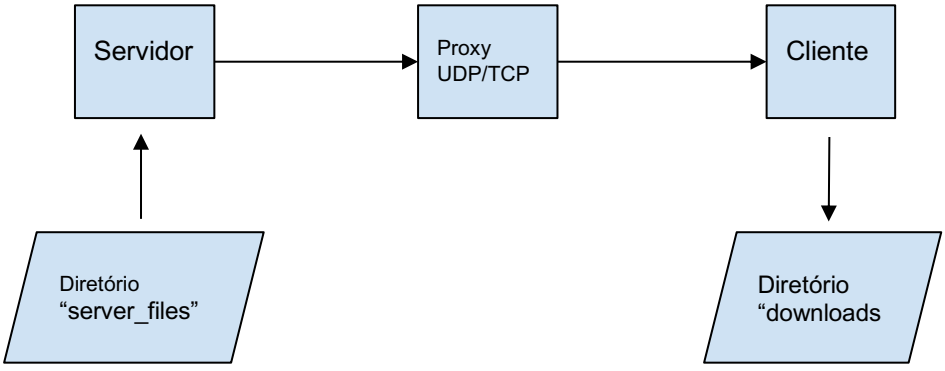

Figura 1: Componentes da aplicação a desenvolver

O servidor deverá estar acessível num determinado endereço e porto (a indicar através da linha de comando, conforme descrito abaixo). Por sua vez, parte-se do princípio que o cliente apenas irá efetuar uma transferência (download) de cada vez.

Ao comunicar com o servidor, o cliente poderá enviar os comandos a seguir indicados:

- LIST: permite ao cliente listar os ficheiros disponíveis no servidor;
- DOWNLOAD <TCP/UDP> <ENC/NOR> <nome>: permite efetuar o *download* do ficheiro indicado, utilizando o protocolo de transporte indicado e se é pretendida ou não encriptação dos dados transferidos;
- QUIT: permite que o cliente termine a ligação ou comunicação com o servidor;

Deverá ser criada, no servidor, uma diretoria para guardar os ficheiros disponíveis para transferência. Esta diretoria é identificada na Figura 1 como "`server_files`". Por sua vez, do lado do cliente a diretoria utilizada para alojar os ficheiros obtidos do servidor é identificada na mesma figura como "`downloads`".

O programa servidor deverá ser executado na linha de comandos do Linux recorrendo à seguinte sintaxe:

`server {porto} {número máximo de clientes}`

Por sua vez, o programa cliente deverá ser executado na linha de comandos de acordo com a seguinte sintaxe:

`cliente {endereço do proxy} {endereço do servidor} {porto} {protocolo}`

Nota: como protocolo a indicar ao cliente poderá considerar apenas o TCP (a utilizar na ligação inicial entre o cliente e o proxy, bem como entre o proxy e o servidor).

De notar que no comando anterior indica-se o endereço do *proxy*, do servidor ao qual a ligação é efetuada (via *proxy*) e o protocolo de transporte a utilizar nas comunicações (UDP ou TCP). O porto é utilizado quer na ligação do cliente ao proxy quer na ligação deste ao servidor final.

O mesmo programa cliente, depois de ter sido efetuado o *download* do ficheiro, deverá imprimir na consola a seguinte informação:

`Nome do ficheiro transferido (`*`download`*` do servidor para o cliente)`
`Total de bytes recebidos`
`Protocolo de transporte utilizado na transferência do ficheiro`
`Tempo total para `*`download`*` do ficheiro`

Proxy para interceção e controlo de comunicações

O objetivo do *proxy*, tal como referido anteriormente, é mediar as comunicações (UDP ou TCP) entre o cliente e o servidor. Desta forma, o *proxy* é uma aplicação na qual as comunicações poderão ser analisadas, e alteradas. O *proxy* é uma das componentes da aplicação a ser criada (ver Figura 1) e deverá implementar as seguintes funcionalidades:

- Visualizar informação sobre a sessão ativa de transferência de dados entre o servidor e o cliente;
- Guardar em ficheiro a informação transferida entre o servidor e o cliente (último ficheiro que o cliente solicitou ao servidor);

- Simular perdas ao nível das comunicações UDP;

De notar que, se as comunicações entre o cliente e o servidor são encriptadas, o *proxy* não deverá ser capaz de as desencriptar, já que não terá acesso às chaves utilizadas para o efeito. Ou seja, nesta situação é normal que a informação analisada e armazenada em ficheiro no *proxy* esteja encriptada. Caso as comunicações sejam legíveis (portanto não encriptadas) a informação armazenada no proxy será igual à transmitida (salvo eventuais perdas no UDP nas comunicações ou introduzidas pelo *proxy*).

A aplicação que implementa o *proxy* deverá implementar as seguintes funcionalidades, na forma de comandos:

- LOSSES <n>: introduz perdas nas comunicações UDP do servidor para o cliente, descartando <n>% dos bytes transferidos. Nota: esta funcionalidade só deverá ser considerada para as comunicações UDP;

- SHOW: Envia para a consola informação sobre a ligação em curso entre o cliente e o servidor (endereço e porto de origem, endereço e porto de destino, protocolo utilizado);

- SAVE: Ativa ou desativa a gravação em ficheiro dos dados (ficheiros) transferidos do servidor para o cliente;

A aplicação *proxy* deverá poder ser colocada a funcionar através do seguinte comando, no qual o porto indicado irá ser utilizado para aceitar comunicações TCP ou UDP, de acordo com os requisitos anteriores:

```
ircproxy {porto}
```

Requisitos de segurança

Tal como referido anteriormente, ao fazer um *download* o cliente tem a opção de ativar a encriptação das comunicações. Mesmo que o cliente não opte por encriptar o *download* do ficheiro, a aplicação deve autenticar mutuamente os intervenientes (o servidor, o cliente e o *proxy*). Assim, os requisitos de segurança a considerar são os seguintes:

- Confidencialidade: as comunicações poderão ser encriptadas, caso o cliente assim o pretenda. Para o efeito deverá utilizar um algoritmo de encriptação simétrica;

- Autenticação: no início da comunicação, cada entidade (cliente, servidor e proxy) deverão autenticar-se mutuamente. Para esse efeito, poderá recorrer a chaves secretas pré-configuradas (PSK, *Pre-Shared Keys*) nas aplicações.

Nota: não será necessário implementar funções criptográficas no trabalho, o objetivo é utilizar uma biblioteca para o efeito, ver biblioteca Libsodium em: https://libsodium.gitbook.io/doc/.

Entrega do Trabalho

- Realização do trabalho: Trabalho individual ou em grupos de dois alunos.

- A entrega do trabalho deverá ser efetuada até dia 15 de Dezembro de 2019, por *upload* no inforestudante do relatório e dos ficheiros com o código fonte da aplicação desenvolvida.

Notas importantes:

- Na realização do trabalho deverá recorrer à linguagem de programação C.

- O relatório deve ser sucinto (no máximo 4 páginas A4), no formato PDF (não serão aceites outros formatos). No relatório deve explicar as opções tomadas na construção da solução e o modo de funcionamento.

- Crie um arquivo no formato ZIP (não serão aceites outros formatos) com todos os ficheiros do trabalho:

    - Todos os ficheiros fonte e de configuração necessários.
    - Não inclua quaisquer ficheiros não necessários para a compilação ou execução do programa (ex. diretórios ou ficheiros de sistemas de controlo de versões)
    - Não serão admitidas entregas por e-mail.

- A defesa final do trabalho é obrigatória para todos os elementos do grupo.

- Todos os trabalhos serão escrutinados para deteção de cópias de código.