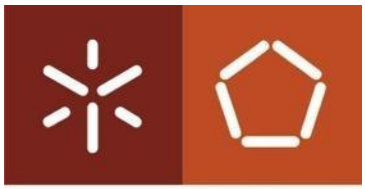

Universidade do Minho

Escola de Engenharia

Mestrado em Engenharia Informática
Engenharia de Serviços em Rede

TP1 - Nível Aplicacional
Conceitos Introdutórios

Grupo 37

Pedro Veloso (pg47578)
Carlos Preto (pg47089)

Braga, Outubro de 2021

Questões e Respostas

Questão 1) As aplicações em rede assentam normalmente em paradigmas cliente-servidor ou peer-to-peer.

a) Explique em que se diferenciam ambos os modelos, salientando o papel das principais entidades envolvidas.

As principais arquiteturas utilizadas por aplicações em rede baseiam-se, normalmente, em arquiteturas *cliente-servidor* ou *peer-to-peer*.

A arquitetura *cliente-servidor* é representada pela existência de dois principais componentes, o cliente e o servidor. Nestas arquiteturas, o servidor é responsável por providenciar ao cliente o conjunto de dados pedidos. Assim, geralmente o servidor tem um endereço IP permanente, para além de estar fixado em grandes *datas centers* de modo a permitir escalar a aplicação.

Por fim, ao contrário do servidor, o cliente pode se desconectar da rede sem causar impacto no funcionamento geral da aplicação, uma vez que os clientes não comunicam diretamente entre si. Assim, o cliente geralmente tem um endereço IP dinâmico e comunica com o servidor através da troca de mensagens na rede, o que permite obter os dados necessários para o funcionamento da aplicação.

Por outro lado, a arquitetura *peer-to-peer*, é caracterizada pela inexistência de servidores. Neste tipo de arquiteturas, os *end systems* comunicam diretamente entre si, o que permite uma escalabilidade automática, uma vez que a adição de novos *peers*, além de aumentar a demanda do serviço, também aumenta a sua capacidade. Este fenómeno dá-se devido ao facto de que nestas arquiteturas quando um *peer* se conecta para requisitar um serviço fornece um outro em troca. Contudo nestas arquiteturas, como os *peers* estão conectados de forma descontínua e os seus endereços IP são dinâmicos é necessário um trabalho de manutenção mais complexo.

b) Enuncie vantagens e desvantagens de cada paradigma e casos de aplicação.

A arquitetura ***servidor-cliente*** encontra-se presente em diversas aplicações, sendo alguns exemplos destas aplicações o *Skype*, o *SSH* e o *Email*. Em termos de vantagens, para estas arquiteturas considera-se que:

- Facilita a administração e atualização da rede, na medida em que o administrador tem controlo total sobre toda a infraestrutura;
- Como apenas utilizadores com permissões acedem aos dados, garante uma maior proteção dos mesmos;
- A centralização dos dados concede-lhes uma maior facilidade de acesso.

Relativamente às desvantagens, considera-se:

- *Overload* do servidor em casos de muitos acessos simultâneos, o que origina uma congestão de tráfico;
- Existência de um único ponto de falha.
- Custo elevado de manutenção.

Por sua vez, o paradigma ***peer-to-peer*** encontra-se presente em aplicações como o *DNS*, o *bitTorrent* e o *Winny* (assistente de pesquisa).

Para estas arquiteturas consideram-se as seguintes vantagens:

- Maior velocidade de download. Como os arquivos estão presentes em vários *peers* é possível dividir a carga de transferência pelos vários *peers*, tornando o download mais eficiente;
- Inexistência de um único ponto de falha, uma vez que existem vários *peers* a partilhar o mesmo ficheiro. Assim, caso um *peer* falhe é ainda possível aceder aos ficheiros.

Relativamente às desvantagens, é facilmente percetível que:

- Aumento da carga computacional para gerir este tipo de arquiteturas. Uma vez que os ficheiros encontram-se distribuídos por diversos *peers* que se estão constantemente intermitentes ou com o endereço IP a variar, torna-se mais difícil gerir o acesso aos mesmos;
- Tem segurança reduzida;
- Dada a inexistência de *data centers*, neste tipo de arquiteturas, torna-se mais suscetível à perda de ficheiros, na medida em que se apenas um *peer* o possuir, caso este se desconecte da rede ou remova o ficheiro, este se extingue da rede.

Questão 2) A Tabela 1 identifica tipos de aplicações amplamente usadas na Internet. Essas aplicações ou serviços apresentam diferente sensibilidade ao comportamento e desempenho da rede em si. Para cada tipo de aplicação (ou serviço), identifique qualitativamente os seus requisitos em termos de débito (*throughput*) necessário, atraso e suas variações (*time sensitive*) e perda de dados (*loss sensitive*).

Dê exemplo concreto de aplicações da sua preferência que encaixem em cada tipo. Complemente a resposta quantificando os parâmetros em análise (referencie as suas fontes de informação).

Tabela 1: Exercício 2

Tipos de Aplicações	Débito (throughput)	Atraso e/ou Jitter (time sensitive)	Perda de dados (loss sensitive)	Aplicações
Web browsing	Flexível	Não	Não tolera perdas	• Firefox • Google Chrome
Multimedia streaming	Inflexível	Sim	Tolera perdas	• Twitch • Youtube
IP Telephony (VoIP)	Flexível	Sim	Tolera perdas	• Zoiper • Bria X-Lite
File transfer/sharing	Flexível	Não	Não tolera perdas	• SHAREit • Bittorrent
Interactive Games	Flexível	Sim	Tolera perdas	• Fortnite • Fifa
Video Conferencing	Inflexível	Sim	Tolera perdas	• Zoom • Skype

Com base nos dados presentes na tabela acima, é possível aprofundar o conhecimento acerca de cada um dos tipos de aplicações referenciados.

- ***Web browsing****:* Este tipo de aplicações é bastante flexível ao débito de informação, visto que irão sempre funcionar com qualquer tipo de velocidades de ligação, obtendo uma experiência mais fluída de navegação para velocidades mais elevadas. Relativamente a atrasos, verifica-se que as aplicações de *Web browsing* não são afetadas pelo *delay* da chegada de pacotes, nem pela variação do *delay* destes. As aplicações de navegação na *internet* não são tolerantes à perda de pacotes.

- ***Multimedia Streaming*** e ***Video Conferencing****:* Ao contrário das aplicações de *Web browsing*, as aplicações de *streaming* de multimédia e de videoconferência necessitam de valores mínimos de débito para que se possa usufruir destas mesmas. Para áudio são necessárias velocidades superiores a 5kbps e para vídeo este valor é ligeiramente maior, aproximando-se dos 10kbps. Também é bastante afetada pelo *delay*, uma vez que a presença de atrasos no *streaming* fará com que o conteúdo apresentado não corresponda ao mais atual. Também se considera que estas aplicações toleram a perda de pacotes, visto que essa mesma perda apenas se traduzirá num impacto negativo durante um curto intervalo de tempo. Para cada uma das aplicações referenciadas, é possível ter uma ideia geral de valores específicos para cada um dos parâmetros. Assim, por exemplo, para ser possível fazer *streaming* de vídeo na *Twitch*[1] é necessária uma velocidade de download e de upload entre 3 a 6Mbps. No caso das aplicações de videoconferência, sabe-se que, por exemplo, para realizar chamadas de áudio no S*kype*[2], é necessária uma velocidade mínima de download/upload de 30kbps. Já para chamadas de vídeo e partilha de ecrã é necessária uma velocidade mínima de download e upload de 128kbps.

- ***VoIP****:* Aplicações do tipo *IP Telephony* são bastante flexíveis ao débito de informação, visto que irão funcionar sempre com qualquer tipo de velocidades de ligação, obtendo sempre uma experiência mais fluída de navegação para velocidades mais elevadas. Atrasos nas chamadas VoIP fazem com que a qualidade da chamada seja gravemente afetada. Neste tipo de aplicações, considera-se aceitável uma perda[3] de 5% dos pacotes, considerando-se por isso que este tipo de aplicações é tolerante à perda de pacotes. Para aplicações VoIP, como no caso do *Zoiper*[4], a largura de rede necessária para fazer chamadas irá depender do tipo de áudio que se pretenda usar, porém, o *bandwidth* mínimo necessário para poder fazer chamadas VoIP é de 8kbps.

- ***File Transfering****:* A transferência de ficheiros é bastante flexível nos débitos de transferência, uma vez que mesmo com uma velocidade de ligação bastante baixa, é na mesma possível descarregar um determinado ficheiro, havendo, no entanto, a desvantagem de essa mesma transferência acarretar um elevado tempo de execução. Este tipo de aplicações não são muito afetadas pelo *delay* na ligação, porém, não tolera

[1] https://frontier.com/resources/internet-speed-for-streaming-twitch

[2] https://support.skype.com/en/faq/FA1417/how-much-bandwidth-does-skype-need

[3] https://www.voipspear.com/posts/10

[4] https://www.zoiper.com/en/support/answer/for/common/104/Audio_codecs

qualquer perda de pacotes, pois, tal como espectável, caso a meio de uma transferência haja a perda de um dado pacote de um dado ficheiro, esse mesmo ficheiro não irá corresponder ao ficheiro original.

- ***Interactive Games**:* Tal como na transferência de ficheiros, os *Interactive Games* funcionam com qualquer tipo de débito de ligação, sendo que quanto maior a velocidade de ligação, mais suave será a experiência durante o jogo. Este tipo de aplicações é bastante sensível a *delays* na rede, fazendo com que caso haja um atraso na entrega de um dado pacote de dados a um dado jogador, esse mesmo jogador seja bastante prejudicado na sua experiência de jogo. Tal como nas aplicações *Multimedia Streaming*, estas aplicações toleram a perda de pacotes, visto que essa mesma perda apenas se traduzirá num impacto negativo durante um curto intervalo de tempo, como, por exemplo, quando as texturas não carregam no tempo devido. No caso do *Fortnite*[5], por exemplo, é necessário um *ping* compreendido entre 10 a 60ms, sendo que para valores superiores haverá um atraso significativo de dados a chegar ao utilizador.

Questão 3) Considere a topologia da Figura 1 onde será distribuído um ficheiro de tamanho X Gbits entre N nodos (*hosts*), Assuma que os débitos de download e upload do nodo i. são respetivamente di e ui . Assuma ainda que:

(i) Os *hosts* estão dedicados à distribuição do ficheiro, i.e. não realizam outras tarefas;

(ii) O núcleo da rede (core) não apresenta qualquer estrangulamento (*bottleneck*) em termos de largura de banda, i.e., qualquer eventual limitação existe nas redes de acesso dos vários ni.

O valor de X deve ser indexado ao identificador de cada grupo de trabalho, i.e., X=IDGrupo/10.

Sabendo que o servidor tem um débito de upload us=1Gbps, e que di =100Mbps, calcule, justificando, o tempo mínimo de distribuição de F pelos N nodos quando N=10, N=100 e N=1000, e para débitos de upload ui de:

a) 1Mbps;

b) 5Mbps;

c) 10Mbps.

Usando os modelos de distribuição:

(i) cliente-servidor;

(ii) peer-to-peer.

Apresente os resultados numa tabela comparativa, bem como o processo de cálculo. Que conclusões pode tirar?

[5] https://www.phoenixinternet.com/the-best-internet-speed-for-gaming-streaming

Dados:

$F = 3.7/10 \text{ Gbits} = 3.7 * 2^{30} \; bits$

$us = 1 \; Gbps = 1 * 10^9 \; bps$

$di = 100 \; Mbps = 100 * 10^6 \; bps$

$X = \frac{37}{10} = 3.7 \text{ Gbits}$

Cálculos:

i)

$D \geq max\{\frac{NF}{us}, \frac{F}{dmin}\}$

$\frac{NF}{us} = \frac{10 * 3.7 * 2^{30}}{1 * 10^9} = 39.728 \, , para \; N = 10$

$\frac{F}{dmin} = \frac{3.7 * 2^{30}}{100 * 10^6} = 39.728$

ii)

$D \geq max\{\frac{F}{us}, \frac{F}{dmin}, \frac{NF}{us + \sum ui}\}$

$\frac{F}{us} = \frac{3.7 * 2^{30}}{1 * 10^9} = 3.973$

$\frac{F}{dmin} = \frac{3.7 * 2^{30}}{100 * 10^6} = 39.728$

$\frac{NF}{us + \sum ui} = \frac{10 * 3.7 * 2^{30}}{1 * 10^9 + 10 * 1 * 10^6} = 39.335 \, , para \; N = 10 \; e \; ui = 1 * 10^6$

Resultados:

Ao aplicar as fórmulas indicadas acima e expandindo os cálculos apresentados para N = 10 e ui=1Mbps, obtém-se os seguintes resultados:

Tabela 2: Exercício 3

	i) Cliente-servidor			*ii) Peer-to-peer*		
	N=10	N=100	N=1000	N=10	N=100	N=1000
A) 1Mbps	39.728	397.284	3972.845	39.728	361.168	1986.422
B) 5Mbps	39.728	397.284	3972.845	39.728	264.856	662.141
C) 10 Mbps	39.728	397.284	3972.845	39.728	198.642	361.168

Ao analisar a tabela 2 percebe-se que a arquitetura *peer-to-peer* é mais escalável, uma vez que, o aumento de utilizadores, apesar de aumentar a demanda aumenta também a capacidade de serviço. Assim, nestas arquiteturas, o aumento de utilizadores não implica um aumento linear do tempo de distribuição de um ficheiro.

Ao contrário desta arquitetura, existe a arquitetura cliente-servidor em que o aumento dos utilizadores cria um *bottleneck* no servidor, deteriorando o desempenho da aplicação. Nestas arquiteturas o aumento de clientes implica um aumento linear do tempo de distribuição de um ficheiro, apresentando assim pior desempenho (em casos de muitos clientes) do que a arquitetura *peer-to-peer*.

Por fim, verifica-se que a taxa de *upload* de um ficheiro por parte do cliente (***ui***) não afeta em nada o tempo de distribuição de um ficheiro na arquitetura cliente-servidor. Este fenómeno dá-se porque nestas arquiteturas os clientes não comunicam entre si. Por outro lado, nas arquiteturas *peer-to-peer* esta taxa influência bastante o tempo para distribuir um ficheiro, uma vez que cada *peer* realiza o *upload* do ficheiro para a rede de modo a partilhar o mesmo com outros *peers*.

Conclusão

Durante a realização deste trabalho, consolidaram-se conceitos relativos às diferentes arquiteturas: *cliente-servidor*, *peer-to-peer*.

A arquitetura *cliente-servidor* é caracterizada pela existência de um único servidor ao qual os clientes se conectam. Esta arquitetura apresenta vantagens e desvantagens, onde algumas das quais foram abordadas neste trabalho.

Por outro lado, foi também abordado aspetos relativos à arquitetura *peer-to-peer*, como por exemplo o seu funcionamento e as suas vantagens/desvantagens.

Por fim foi realizado uma comparação entre as diferentes arquiteturas, através de uma estimativa dos tempos de distribuição de um ficheiro nas duas arquiteturas.