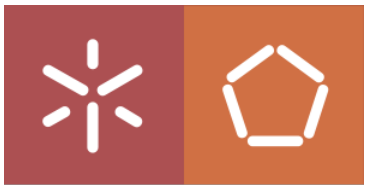

Universidade do Minho
Escola de Engenharia

Mestrado em Engenharia Informática

Engenharia de Serviços em Rede

Trabalho Prático nº1

Nível Aplicacional: Conceitos Introdutórios

Ano Letivo 2021/2022

Grupo 6 - PL1

Ana Filipa Pereira PG46978
Carolina Santejo PG47102
Raquel Costa PG47600

Conteúdo

1 Questões e Respostas **3**
1.1 Questão 1 3
1.2 Questão 2 5
1.3 Questão 3 9

1 Questões e Respostas

1.1 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.

R: O paradigma cliente-servidor apresenta uma arquitetura de aplicação distribuída na qual todos os clientes se conectam a um servidor centralizado, sendo que este se encontra sempre à escuta de novos pedidos. O server é responsável por gerir e disponibilizar recursos a diversos clientes sendo que deve atender todos os seus pedidos e responder aos mesmos com os dados solicitados. Além disso, é de notar que o endereço IP do servidor é estático. Os clientes iniciam pedidos ao servidor e aguardam até receber resposta. É também de realçar que estes poderão ter um endereço IP dinâmico e não comunicam diretamente entre si.

Por outro lado, o modelo peer-to-peer trata-se de uma arquitetura de redes na qual os serviços e dados podem ser partilhados sem a necessidade de um servidor central. Todos os nós da rede são iguais e podem desempenhar as mesmas funções, nomeadamente, efetuar e responder a pedidos. Sendo assim, cada um atua tanto como cliente ou servidor. Estes peers possuem um endereço IP dinâmico.

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

R: Uma das grandes vantagens da arquitetura cliente-servidor é o facto de esta ser centralizada. Os recursos estarem guardados num único ponto da rede facilita os processos de manutenção, correção de erros e de atualização da informação, uma vez que estes apenas são feitos no server central e não em todos os hosts da rede. Esta centralização contribui também para a segurança, dado que facilita a implementação de mecanismos de controlo de acesso, como por exemplo, o uso das credenciais username e password. Isto permite que apenas hosts autorizados tenham acesso aos recursos da rede. Além disto, outra vantagem deste paradigma é a escalabilidade uma vez que é possível adicionar novos servidores e/ou clientes à rede, sem que esta sofra grandes interrupções. No entanto, é importante realçar que esta escalabilidade só é possivel até um certo número de clientes , uma vez que o servidor deixa de ser capaz de responder a todos os pedidos.

Por outro lado, esta arquitetura apresenta também algumas desvantagens. O facto de ser centralizada leva a que o servidor central seja um ponto de falha. Se este parar de funcionar, os clientes deixam de poder fazer pedidos. Além disto, a congestão de tráfego é também outra desvantagem. Caso haja um elevado número de clientes a efetuar pedidos em simultâneo para o mesmo servidor

central, este pode ficar mais lento e por isso torna-se um ponto de contenção de desempenho. Alguns dos casos de aplicação mais conhecidos que utilizam a arquitetura cliente-servidor são a Web, e-mail, FTP e Telnet.

No caso do paradigma peer-to-peer, a sua arquitetura descentralizada apresenta algumas vantagens nomeadamente a tolerância a falhas. O facto de não existir um servidor central, permite que não haja um único ponto de falha, e por isso, mesmo que um dos nós fique indisponível, a rede continuará funcional. Além disto, o custo para criar e gerir uma rede deste tipo é relativamente baixo, visto que não é necessário uma configuração centralizada. O modelo P2P apresenta não só alta escalabilidade, sendo que a adição de novos peers aumenta a capacidade da rede, mas também uma maior largura de banda o que permite responder a mais pedidos de clientes em simultâneo sem afetar o desempenho da mesma.

No entanto, este modelo apresenta também algumas desvantagens. Em primeiro lugar, e ao contrário do paradigma cliente-servidor, não há outro método de segurança a não ser permissões de acesso. Além disto, a descentralização torna mais difícil o processo de backup e de atualização dos dados uma vez que estes se encontraram distribuídos pelos vários peers da rede. Como um nó da rede pode ser acedido por outros isto leva a que o utilizador possa sentir uma perda de performance. Alguns dos casos de aplicação mais conhecidos que utilizam a arquitetura P2P são o BitTorrent, Skype e Xunlei.

1.2 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).

Tipos de Aplicações	Débito (throughput)	Atraso e/ou Jitter (time sensitive)	Perda de dados (loss sensitive)	Aplicações
Web browsing	Elástico	sim	não tolerante	Google Chrome
Multimedia streaming	audio:5kbps-1Mbps video:10kbps-5Mbps	sim	tolerante a perdas	Netflix, HBO
IP Telephony (VoIP)	10kbps-5Mbps	sim	tolerante a perdas	WhatsApp
File transfer/sharing	Elástico	não	não tolerante	Dropbox
Interactive Games	até 10kbps	sim	tolerante a perdas	Fortnite
Video Conferencing	10kbps-5Mbps	sim	tolerante a perdas	Skype, Zoom

Tabela 1: Tabela com valores generalizados.

Fonte: "Computer-Networking-A-Top-Down-Approach-7th"

Tipos de Aplicações	Débito (throughput)	Atraso e/ou Jitter (time sensitive)	Perda de dados (loss sensitive)	Aplicações
Web browsing	0.2-0.5Mbps	<100ms	não tolerante	Google Chrome
Multimedia streaming	audio:5kbps-1Mbps video: SD-3Mbps HD-5Mbps UHD-25Mbps	<30ms	tolerante a perdas ($< 1\%$)	Netflix
IP Telephony (VoIP)	10kbps-5Mbps	<10ms	tolerante a perdas ($< 5\%$)(ROSS,2007)	WhatsApp
File transfer/sharing	Elástico	não	não tolerante	Dropbox
Interactive Games	[14-64]kbps	<70ms	$\pm 0\%$	League of Legends
Video Conferencing	HD - 600kbps 720p - 1.2Mbps 1080p - 1.8Mbps	<20ms	tolerante a perdas ($0.1\% - 2\%$)	Zoom

Tabela 2: Tabela com valores específicos para cada aplicação.

- **Web Browsing**

  No caso do Google Chrome verifica-se um débito que ronda os 0.2Mbps e os 0.5Mbps além de se observar que um atraso é considerado aceitável se for inferior a 100ms. Nesta situação não é tolerada a perda de pacotes uma vez que isto levaria a que uma página Web não mostrasse todos os seus conteúdos.

  Fonte: https://www.igvita.com/posa/high-performance-networking-in-google-chrome/

- **Multimedia streaming**

  No serviço de streaming da Netflix verifica-se que quanto maior for qualidade de vídeo maior será o débito mínimo necessário, além de que um atraso é considerado aceitável se for inferior a 30ms. Neste caso, existe alguma tolerancia a perdas desde que estas sejam não sejam superiores a 1%, pois isto levaria a que o utilizador não tivesse uma experiência visual ininterrupta.

  Fonte: https://www.ringcentral.com/us/en/blog/what-is-jitter/

- **IP Telephony (VoIP)**

  No caso do WhatsApp verifica-se um débito que ronda os 10kbps e os 5Mbps além de se observar que um atraso é considerado aceitável se for inferior a 10ms. Nesta situação é tolerada a perda de pacotes desde que estas sejam não sejam superiores a 5%, pois isto levaria a que o utilizador ouvisse cortes durante uma chamada de aúdio.

  Fonte: https://www.teleco.com.br/tutoriais/tutorialvoipconv2/pagina_2.asp;

- **File transfer/sharing**

  No caso da transferência de ficheiros, não foi possível obter dados concretos acerca de alguma aplicação, no entanto, de forma geral concluiu-se que apresenta um débito muito variável. Por sua vez, este tipo de aplicação não é sensivel a atrasos, uma vez que não é necessária uma resposta em tempo real. Neste tipo de aplicação não pode existir perda de dados, uma vez que é indispensável que o ficheiro seja recebido na totalidade.

- **Interactive Games**

  A aplicação em análise para o caso dos *Interactive Games* é o jogo *League of Legends*. Ao longo de um jogo normal no modo principal é estimado que um jogador obtenha um *throughput* de download e upload que ronda os 64 e 14 kbps respetivamente.
  Em relação à latência, nos jogos é costume usar o ping como forma de medida. De acordo com as fontes e pela experiência, um ping é aceitável até aos 70ms, a partir daí, já são valores que tornam a jogabilidade quase impossível ou então com muitos problemas e defeitos. Um intervalo que compreende valores entre os 20ms e os 70ms, considera-se normal, sendo que abaixo dos 20 são valores bastante bons que permitem uma boa jogabilidade.
  Já no caso da perda de dados, verificou-se que ocorre quando os *data packets* não chegam ao computador ou aos servidores da Riot (empresa de jogos responsável pelo jogo em questão). Os pacotes em falta precisam de ser retransmitidos, o que provoca um delay em relação ao processamento da informação, resultando assim na existência de lag durante o jogo. Portanto, a perda de dados deverá ser o mais próximo possível de 0%, de modo a tornar a jogabilidade aceitável.

  Fontes: https://www.esports.net/wiki/guides/ping-league-of-legends/;
  https://oce.learnwithleague.com/knowledgebase/what-is-lols-typical-data-and-bandwidth-usage/

- **Video Conferencing**

  No caso do Video Conferencing verifica-se que quanto maior for qualidade de vídeo maior será o débito mínimo necessário, além de que um atraso é considerado aceitável se for inferior a 20ms. Nesta situação é tolerada a perda de pacotes desde que estas sejam não sejam superiores a 2%, pois isto levaria a que o utilizador não tivesse uma experiência de audio e video fluída.

  Fonte: https://support.zoom.us/hc/en-us/articles/202920719-Meeting-and-phone-statistics;
  https://www.reviews.org/internet-service/how-much-data-does-zoom-use/;
  https://www.techrepublic.com/blog/data-center/is-your-network-ready-to-handle-videoconferencing/

1.3 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; e (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 e 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?

$$X = \frac{7}{10} = 0,7$$

N nodos / Débitos de upload u_i	N = 10	N = 100	N = 1000
1Mbp	7.516s	75.16s	751.6s
5Mbps	7.516s	75.16s	751.6s
10Mbps	7.516s	75.16s	751.6s

Tabela 3: Client-Server - Tempo mínimo de distribuição de F

$$D_{C-S} \geq max\{\frac{N \times 0.7 \times 2^{30}}{1 \times 10^9}, \frac{0.7 \times 2^{30}}{100 \times 10^6}\}$$

N nodos / Débitos de upload u_i	N = 10	N = 100	N = 1000
1Mbps	7.516s	68.329s	375.810s
5Mbps	7.516s	50.108s	125.270s
10Mbps	7.516s	37.581s	68.329s

Tabela 4: Peer-to-peer - Tempo mínimo de distribuição de F

$$D_{P2P} \geq max\{\frac{0.7 \times 2^{30}}{1 \times 10^9}, \frac{0.7 \times 2^{30}}{100 \times 10^6}, \frac{N \times 0.7 \times 2^{30}}{1 \times 10^9 + N \times u_i}\}$$

Os resultados obtidos permitem obter as seguintes conclusões:

No caso do modelo cliente-servidor, verifica-se um crescimento linear do tempo mínimo de distribuição de F à medida que o número de clientes aumenta (N). Esta conclusão advém do facto de que quando o número de clientes aumenta 10 vezes, o tempo mínimo de distribuição de F aumenta na mesma proporção. Por outro lado, o débito de upload de cada cliente não tem qualquer impacto, tal como se verifica na tabela, uma vez que os clientes, nesta arquitetura, só efetuam pedidos e recebem dados.

Já na arquitetura peer-to-peer, e de acordo com os resultados obtidos, é possível verificar que quando o número de peers é apenas 10, existe um 'bottleneck' ao nível do servidor, o que faz com que o débito de upload (u_i) não tenha impacto. Por outro lado, quando o número de peers é 100 ou 1000, é possível observar que há uma diminuição do tempo mínimo de distribuição de F com o aumento do débito de upload. Isto acontece, uma vez que, já não é apenas o servidor a distribuir o ficheiro mas também os vários peers, porque estes já têm em sua posse os chunks do ficheiro enviado pelo servidor, que inicialmente só este tinha. É de notar que neste modelo os clientes não só efetuam pedidos como também respondem, daí o seu débito de upload ter impacto nos resultados obtidos. Tal como se pode ver na Figura 1, estes valores apresentam um comportamento aproximadamente logarítmico. As funções logarítmicas inicialmente tendem a ter um comportamento bastante parecido com o das lineares, daí este tipo de função ser representativo da escalabilidade desta arquitetura. Daí a coluna de N=10, em ambas as tabelas terem os mesmos valores de tempo mínimo de distribuição de F.

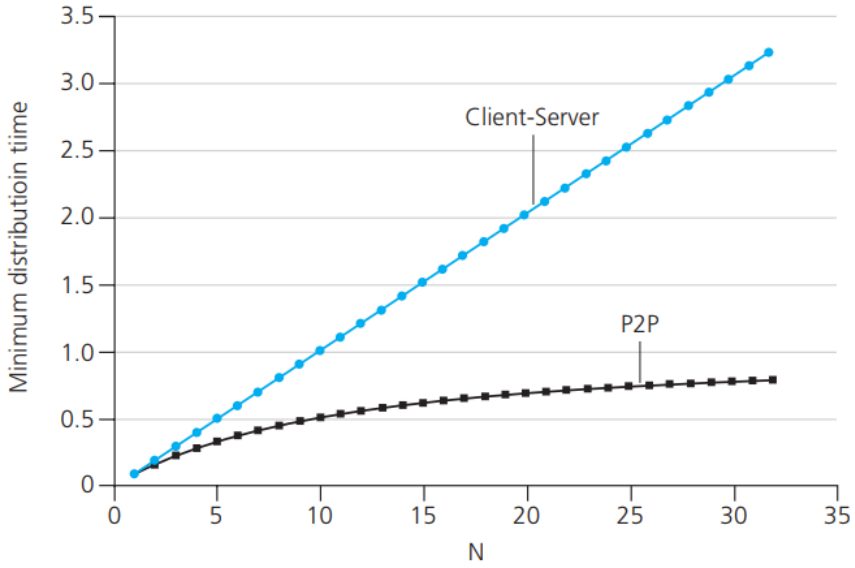

Figura 1: Gráfico de escalabilidade