

Trabalho de aprofundamento AP2

André Patacas, Gil Teixeira



Aplicação para o cálculo de Largura de Banda e de Latência

DETI

André Patacas, Gil Teixeira
(93357) andrepatacas@ua.pt, (88194) gilteixeira@ua.pt

9 de Abril de 2019

Conteúdo

Resumo

Este relatório pretende descrever uma aplicação desenvolvida para calcular a largura de banda e a latência da máquina onde a aplicação se encontra a correr. Calculam-se estes valores para um determinado servidor ou para um conjunto, de cardinalidade especificável, de servidores de um país também este especificável. No final a aplicação cria um relatório, (*report.csv*) , em csv, e assina-o com uma chave privada, (*key.priv*), a ser fornecida pelo utilizador.

Capítulo 1

Introdução

A aplicação foi desenvolvida em python3 no âmbito da disciplina de Laboratórios de Informática, no ano letivo 2018/2019. A adicionar às especificações básicas pedidas, segundo o guião sobre regras do segundo trabalho de aprofundamento, construiu-se ainda suporte para pydocs, para haja uma explicação mais detalhada de cada classe e método do nosso projeto. O programa foi escrito com base em test driven development (Capítulo 2) e, como tal, elaborou-se um esqueleto do programa que se pretendia, seguidos pelos testes unitários e finalmente, por vários updates a ambos até chegar ao estado em que a aplicação se encontra. Finalmente é demonstrado em detalhe um exemplo de utilização da aplicação (??).

Capítulo 2

Metodologia

1. Criar o esqueleto da aplicação;
2. Criar o primeiro teste (*test_client*) de forma a que, a cada método que é construído, se possa testar imediatamente se esse método cumpre exatamente com o que estava especificado;
3. Criar o pydoc para a aplicação e para os testes;
4. Ajustar os métodos de forma a que a que passem a todos testes;
5. Testar o programa manualmente e com testes unitários;
6. Iterar o processo de debugging e correção de erros.

Capítulo 3

Aplicação de Speed Test

3.1 client.py

A forma de utilizar este programa está descrita em detalhe na Subseção 3.1.2. Toda a descrição feita neste relatório remete na mesma para a documentação criada a quando do desenvolvimento da aplicação, em pydoc.

3.1.1 main

Ao correr a aplicação a ordem pela qual os métodos são chamados é a seguinte:

1. `load_server()` - Subseção 3.1.16;
2. `validate()` - Subseção 3.1.3;
3. `run_tests()` - Subseção 3.1.4;
4. `report()` - Subseção 3.1.10;
5. `create_signed_document` - Subseção 3.1.11.

3.1.2 usage()

Este método imprime a mensagem de erro passada como argumento e imprime a ajuda para utilização da aplicação. No campo *option* pode usar *-v* para entrar em modo verbose.

Argumentos: *message* (str).

Retorna: *None*.

3.1.3 validate()

Este método trata da validação dos argumentos passados pela variável `sys.argv`.

Argumentos: *None*.

Retorna: *None*.

3.1.4 run_tests()

Este método serve para calcular a largura de banda e latência da conexão a um *num* de servidores num país, ou a um server com o id passado por argumento. Nota: se o terceiro argumento for um id, a função realizará *num* testes a esses servidor, se for um país, fará *num* testes usando a função Subseção 3.1.5 e se não foi passado terceiro argumento realiza um teste random (Subseção 3.1.7).

Argumentos:

1. *interval*: intervalo de tempo entre cada teste realizado;
2. *num*: número de testes a realizar;
3. *id_or_country*: país (str) ou id (int) de um server;
4. *option*: -v se pretender correr a aplicação em modo *verbose*.

Retorna: objeto *SpeedTestResult* com as informações relativas aos resultados do teste.

3.1.5 country_test()

Este método serve para calcular a largura de banda e latência da conexão a um servidor aleatório do país passado como argumento.

Argumentos: (str) *target_country*.

Retorna: objeto *SpeedTestResult* com as informações relativas aos resultados do teste.

3.1.6 id_test()

Este método serve para calcular a largura de banda e latência da conexão a um servidor com o id passado como argumento.

Argumentos: (int) *target_id*.

Retorna: objeto *SpeedTestResult* com as informações relativas aos resultados do teste.

3.1.7 random_test()

Este método serve para calcular a largura de banda e latência da conexão a um servidor random.

Argumentos: *None*.

Retorna: objeto *SpeedTestResult* com as informações relativas aos resultados do teste.

3.1.8 calc_download()

Cálculo de largura de banda.

Este método pede ao *target_server* um download de 100 mb. Durante 10 segundos é feito download dos dados enviados pelo mesmo. No final dos 10 segundos,

se o download tiver sido superior a 10mb regista, caso contrário, discarta este servidor.

Argumentos: *target_server* (dicionário com informação sobre o target server).

Retorna: (float) $1/time_download_1mb$

3.1.9 calc_latency()

Cálculo da latência.

Este método troca dez *PING-PONG* com o *target_server* e calcula o tempo médio em milisegundos entre estas trocas.

Argumentos: target server (dicionário com informação sobre o target server).

Retorna: (int) *average_trade_time* em ms.

3.1.10 report()

Este método vai gerar um *test_report* baseado numa lista de objetos *SpeedTestResult* passados como argumentos.

Argumentos:

1. List[objeto *SpeedTestResult*];
2. *report_name* (str) - nome do ficheiro a ser gerado.

Retorna: *None*. O ficheiro *test_report* será gerado

3.1.11 create_signed_document

Este método gera um *signature file* assinando o *report* com a chave privada no *key_path* especificada. A chave tem 128 bits e o texto é assinado de 16 em 16 caracteres, por isso deve ser verificado da mesma forma (ver Subseção 3.1.11).

Argumentos:

1. *key_path* (str): O *path* para a localização da chave;
2. *report_name* (str): Nome do *report* a ser assinado;
3. *signature_name* (str): Nome do *signature file* que será gerado.

Retorna: *None*.

3.1.12 log()

Este método imprime a mensagem passada como argumento, com a cor passada como argumento.

Argumentos:

1. *message* (str);
2. *colour* (str).

Retorna: *None*.

3.1.13 log_error()

Este método chama Subseção 3.1.12 com a mensagem igual à passada como argumento mas com cor vermelho.

Argumentos: *message* (str).

Retorna: *None*.

3.1.14 log_warning()

Este método chama Subseção 3.1.12 com a mensagem igual à passada como argumento mas com cor amarela.

Argumentos: *message* (str).

Retorna: *None*.

3.1.15 log_verbose()

Este método chama Subseção 3.1.12 com a mensagem igual à passada como argumento mas com cor verde se o modo *verbose* estiver ativado.

Argumentos: *message* (str).

Retorna: *None*.

3.1.16 load_server()

Este método lê o ficheiro *"servers.json"* e cria um dicionário global com a lista de servidores.

Argumentos: *None*.

Retorna: *None*.

3.2 test_client

Este programa é constituída por métodos que são testes unitários aos da aplicação principal (Seção 3.1). Lista de funções com testes unitários:

1. test_calc_download() : Subseção 3.1.8;
2. test_calc_latency(): Subseção 3.1.9;
3. test_country_test(): Subseção 3.1.5;
4. test_create_signed_document(): Subseção 3.1.11;
5. test_id_test(): Subseção 3.1.6;
6. test_random_test(): Subseção 3.1.7;
7. test_report(): Subseção 3.1.10;
8. test_run_test(): Subseção 3.1.4.

3.3 speed_test_result

Este programa serve para criar objetos SpeedTestResult que têm, cada um, as informações respectivas a um teste. Tem apenas um construtor e um método:

3.3.1 Construtor

O construtor da classe cria um objeto com os parametros passados como argumentos: **Argumentos:**

1. *server_id* (int);
2. *download_speed* (float);
3. *latency* (int);

3.3.2 getObjDict

Este método devolve um dicionário com os resultados do teste relativo ao objeto. O último elemento do dicionário é o resultado do processo de hashing por SHA256 dos atributos anteriores concatenados.

Argumentos: *None*.

Retorna: *testResult* (dict).

Capítulo 4

Análise de um exemplo e Conclusão

4.1 Exemplo de utilização 1

Com exemplo ir-se-á correr a aplicação Seção 3.1 com os argumentos:

1. *interval* = 5 (segundo);
2. *num* = 3 (testes);
3. *id_or_country* = Portugal;
4. *option* = -v (*verbose*).

Ao correr a aplicação com estes argumentos, segundo a Subseção 3.1.2, vêm os seguintes resultados??:

Criando-se dois novos ficheiros na pasta onde está a aplicação:

- report.sig, contendo uma assinatura do relatório pela chave privada fornecida (*key.priv*).
- report.csv, um ficheiro **csv!** (**csv!**) com os resultados dos três testes efetuados:

Contador	Id Do Servidor	Data e Hora no Formato ISO	Latencia	Largura de Banda	Check
1	9729	2019-04-19 23:04:48.745678	13	1.3335593613565648	aa8c139784e517d2f79a785fa224767b8714d494d8a4ede60a755c184019763e
2	9729	2019-04-19 23:05:03.921081	14	1.682293124975788	d12a70a22302919711a5f9233340b27b6cfeb4d2ac81ed2f3575048bca51012a
3	1902	2019-04-19 23:05:19.024825	6	1.882504664392915	2f1b6bc5511c4db3c74143fc903bf11a82dad81d0dcbf6b2fb3927af85542c1e

4.2 Exemplo de utilização 2

Com exemplo ir-se-á correr a aplicação Seção 3.1 com os argumentos:

```

gil@gil-teixeira:~/Desktop/labi-ap02$ python3 client.py 5 3 Portugal -v
Starting Test Phase
  Starting a Network Test to Server in Portugal
    Starting Download Speed Test to porto.speedtest.net.zon.pt
    Download Speed Test done to porto.speedtest.net.zon.pt: 1.3364588900746175MB/s
    Starting Latency Test to porto.speedtest.net.zon.pt
    Latency Test done to porto.speedtest.net.zon.pt: 15ms
  Starting a Network Test to Server in Portugal
    Starting Download Speed Test to speedtest1.meo.pt
    Download Speed Test done to speedtest1.meo.pt: 1.700130344277497MB/s
    Starting Latency Test to speedtest1.meo.pt
    Latency Test done to speedtest1.meo.pt: 12ms
  Starting a Network Test to Server in Portugal
    Starting Download Speed Test to speedtest3.meo.pt
    Download Speed Test done to speedtest3.meo.pt: 1.904441075354276MB/s
    Starting Latency Test to speedtest3.meo.pt
    Latency Test done to speedtest3.meo.pt: 11ms
Test Phase Ended
Starting Report Creation Phase
Report Created Starting to Sign the Report
Report Signed

```

1. *interval* = 1 (segundo);
2. *num* = 20 (testes);

Ao correr a aplicação com estes argumentos, segundo a Subsecção 3.1.2, vêm os seguintes resultados:

Criando-se dois novos ficheiros na pasta onde está a aplicação:

- report.sig, contendo uma assinatura do relatório pela chave privada fornecida (*key.priv*).
- report.csv, um ficheiro **csv**! com os resultados dos vinte testes efetuados:

Comador	Id Do Servidor	Data e Hora no Formato ISO	Latencia	Largura de Banda	Check
1	3787	2019-04-23 23:15:11.814397	72	1.2430874408544983	e3303529c750ab6fb663657bc981911ef7db2f3d467ae11ab6a19d17ac3a1ed
2	6907	2019-04-23 23:15:25.371232	200	1.453856874134937	0dce75c41407aad02ac0552bced5f5d3340e6849815c8a4e4ad1caed55edd0c
3	12322	2019-04-23 23:15:40.323813	301	1.6721249958527682	602ca564434909c5b1471122b51d0a8cc6311c2499ac0cc8a0d51b5bd76aa7
4	13606	2019-04-23 23:15:55.476764	300	0.3002715865565481	69c5586b72267a7629471b90506d27bc4c56b586572129603657bc0ffc4becced
5	21873	2019-04-23 23:16:10.429150	305	0.6919384792792023	becdbf5a489547184f0ce5dce68f7a9c7ef3d4766c251b524935b5744ace817b
6	20978	2019-04-23 23:16:22.785171	96	1.8346690686226794	06727d7d25a5a259931278f2d543a5fdad855d8f32c602eace5cca41c6bf4ac
7	10728	2019-04-23 23:16:38.828644	301	0.210595319242981	58fd45509cbe10a36352c9f4770f5ca29bf5f67d27732f8a47686ec97dcf6b
8	3482	2019-04-23 23:16:51.987782	164	1.8180169922073217	e0e9678472c4405a8a042c29dd20b71e5d8c6058dc353aca592f0c9cfeb6c5
9	7035	2019-04-23 23:17:05.421876	188	1.9264675928248938	37b6d7f555e0e856812e3f524067dd37d642c4f234d7d3300bce9ad296b3a9f7
10	9566	2019-04-23 23:17:20.374519	298	1.8944690340216246	0ac63397a48e204307f91ffdd92b95acbf50d7910895f5d7edc66ba5c544f0c
11	3555	2019-04-23 23:17:32.72854	94	2.0166120335557522	abd80b26ec548bf56caab155544ca524bb516072dbb9d2cfea28f738354a7f67c
12	2845	2019-04-23 23:17:47.773268	324	1.8117547004538457	f001cd88c662dde20635deae16c37504354ad263ac5b8ce04a08eb595f057d21
13	10230	2019-04-23 23:17:59.925717	87	2.057615821157388	88a60195bd227a14ffb815da263d622465a86f983f2777f1c453c28c9fadf84b
14	1609	2019-04-23 23:18:13.560844	200	2.063724927823838	dc7f4eb37eb5d6d115c4de242c65c5066c8710c33e46386d224a65602ad2bce9
15	17427	2019-04-23 23:18:28.513233	299	1.1734104673206632	f39c6fba23407b29c7d4df5c0391d8d505b9455aac8cfd2482324285914c28
16	19534	2019-04-23 23:18:42.161013	192	1.1743414579010149	76c3d89180d78aa4f50ebb7d30ccbb105364cc910ac3ccf5de84d07c37c25db4f
17	10420	2019-04-23 23:18:54.102702	75	2.132581818908019	5c563bf487f06307a53d79d78b486b82afdc08d687c634969c9c7cd61fdd5b
18	19866	2019-04-23 23:19:08.813828	300	1.974450059487398	c320d4a4da6fa85634760c6d7e0aa7b786d9c74b4ed5a27fb39dd18a979944ab
19	16635	2019-04-23 23:19:23.506009	305	1.8385825253261963	027bb4e49c96ca3abe104c74adb30f18a5c3c0a5d8c52c8d6c7a221259e9d1c7
20	1369	2019-04-23 23:19:38.659193	301	0.3467424459681414	18b151603ce853562f6041b0c1a59d3acbc5c0eb0260bb86c90fac52ed3c8f88

4.3 Conclusão

A aplicação tem também bons mecanismos de *handling* de exceções, de forma a que as várias exceções que possam surgir possam ser mostradas.

(Nota: algumas mensagens sobre exceções só são visíveis em modo *verbose* (ver Subseção 3.1.2).

Acrónimos

CSV Comma Separated Values