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 relatõrio, (*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, elabourou-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 4).

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 construido, 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 variávle 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. *inteval*: 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 constituida 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 respetivas 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:

```
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
```

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 Comma Separated Values (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	1.6822931249757888	d12a70a22302919711a5f92333340b27b6cfeb4d2ac81ed2f3575048bca51012a
3	1902	2019-04-19 23:05:19.024825	6	1.8825046643929215	2f1b6bc5511c4db3e74143fc903bff1a82dad81d0dcbf6b2fb3927af8554 2c1e

4.2 Exemplo de utilização 2

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

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

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