

INGENIERÍA DE SERVIDORES (2021-2022)
GRADO EN INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE GRANADA

Memoria Práctica 4

Alberto Llamas González

5 de enero de 2022

Índice

1 Ejercicio Phoronix	3
1.1 Instalación de Phoronix en CentOS	3
1.2 Instalación de Phoronix en UbuntuServer	4
1.3 Ejecución de Benchmarks en los S.O	5
1.3.1 Ejecución del benchmark Sudokut	5
1.3.2 Ejecución del benchmark Git	6
2 Ejercicio JMeter	8
2.1 Instalación de Docker en Ubuntu Server	9
2.2 Instalación de la aplicación para el test con JMeter	11
2.3 Instalación de JMeter en Host	12
2.4 Configuración del test con JMeter	13

1. Ejercicio Phoronix

Phoronix es una plataforma que permite ejecutar un conjunto de benchmarks bajo la agrupación openbenchmarking.org. La aplicación puede instalarse a través de los gestores de paquetes ya vistos en los guiones anteriores. Una vez que haya indagado sobre los benchmarks disponibles, seleccione como mínimo dos de ellos y proceda a ejecutarlos en Ubuntu y CentOS. Comente las diferencias.

En primer lugar, instalaremos Phoronix tanto en CentOS como en Ubuntu Server.

1.1. Instalación de Phoronix en CentOS

Antes de instalar Phoronix, necesitamos instalar las dependencias necesarias.

```
$ yum install wget php-cli php-xml bzip2
```

Una vez instaladas dichas dependencias, nos descargamos el programa del repositorio oficial, descomprimimos el fichero e instalamos el script para instalarlo.

```
1 $ wget https://phoronix-test-suite.com/releases/phoronix-test-suite
   -8.4.1.tar.gz
2 $ tar xvfz phoronix-test-suite-8.4.1.tar.gz
3 $ cd phoronix-test-suite
4 $ ./install-sh
```

Para comprobar que lo hemos instalado correctamente ejecutamos

```
1 $ phoronix-test-suite system-info
```

para que nos muestre la información del sistema CentOS en el que nos encontramos.

```
albertollamasgonzalez - albertolg@localhost:~ -- ssh -p 22022 albertolg@192.168.56.110 -- 88x40
Phoronix Test Suite v8.6.0
AN OUTDATED VERSION OF THE PHORONIX TEST SUITE IS INSTALLED.
THE VERSION IN USE IS 8.6.0 (8600), BUT THE LATEST IS PTS-CORE 10800.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

System Information

:
Core Count: 1
Extensions: SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE
Cache Size: 4096 KB

:
Screen: VMware SVGA II
2048x2048

:
BIOS Version: Oracle VirtualBox v1.2
Chipset: VirtualBox
Intel 440FX 82441FX PMC
Audio: Intel 82801AA AC 97 Audio
Network: 2 x Intel 82540EM

:
818MB

:
File-System: 2 x 9GB VBOX HDD
Mount Options: xfs attr2 inode64 noquota relatime rw seclabel
Disk Scheduler: MQ-DEADLINE

:
Kernel: CentOS Linux 8
4.18.0-193.el8.x86_64 (x86_64)
System Layer: Oracle VMWare
Security: SELinux
+ KPTI
+ usercopy/swaps barriers and __user pointer sanitization
+ Full generic retpoline STIBP: disabled RSB filling
+ PTE Inversion

[albertolg@localhost ~]$
```

1.2. Instalación de Phoronix en UbuntuServer

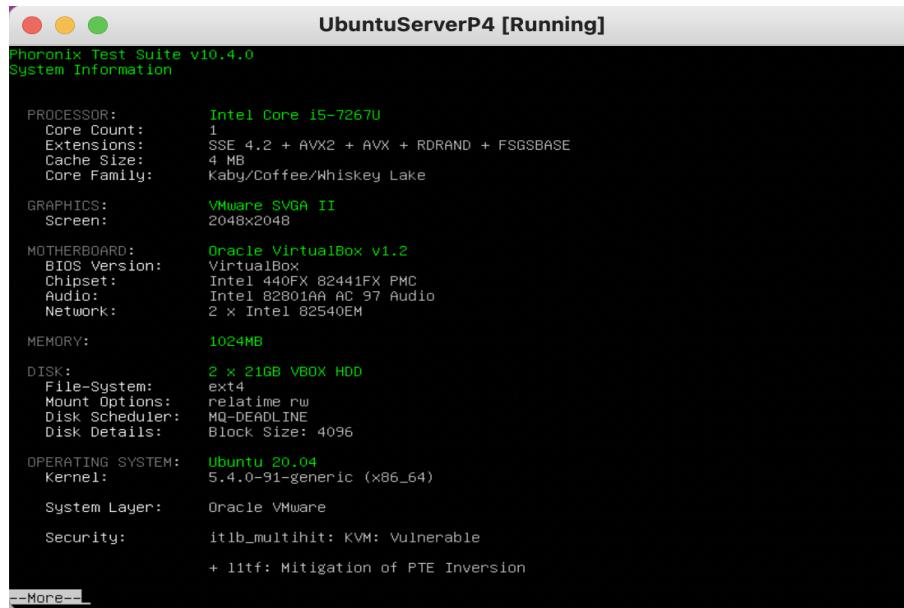
Instalamos ahora Phoronix en Ubuntu Server. Hacemos lo mismo que en CentOS

```
1 $ wget wget http://phoronix-test-suite.com/releases/repo/pts.debian/files/phoronix-test-suite_8.4.1_all.deb  
2 $ sudo dpkg -i phoronix-test-suite_8.6.0_all.deb  
3 $ sudo apt -f install
```

El último comando lo utilizamos para instalar las dependencias que nos faltan (php-cli,php-xml).

```
alberto@ubuntu:~$ sudo dpkg -i phoronix-test-suite_10.4.0_all.deb  
Selecting previously unselected package phoronix-test-suite.  
(Reading database ... 71969 files and directories currently installed.)  
Preparing to unpack phoronix-test-suite_10.4.0_all.deb ...  
Unpacking phoronix-test-suite (10.4.0) ...  
dpkg: dependency problems prevent configuration of phoronix-test-suite:  
  phoronix-test-suite depends on php5-cli | php5-cl; however:  
    Package php5-cl is not installed.  
    Package php5-cl is not installed.  
  phoronix-test-suite depends on php5-cl | php-xml; however:  
    Package php5-cl is not installed.  
    Package php-xml is not installed.  
  
dpkg: error processing package phoronix-test-suite (--install):  
  dependency problems - leaving unconfigured  
Processing triggers for mime-support (3.64ubuntu1) ...  
Processing triggers for man-db (2.9.1-1) ...  
Errors were encountered while processing:  
  phoronix-test-suite  
alberto@ubuntu:~$ sudo apt -f install  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
Correcting dependencies... Done  
The following additional packages will be installed:  
  php5-cll php-common php-xml php7.4-cll php7.4-common php7.4-json php7.4-opcache php7.4-readline  
  php7.4-xml  
Suggested packages:  
  php-pear  
The following NEW packages will be installed:  
  php5-cll php-common php-xml php7.4-cll php7.4-common php7.4-json php7.4-opcache php7.4-readline  
  php7.4-xml  
0 upgraded, 9 newly installed, 0 to remove and 210 not upgraded.  
1 not fully installed or removed.  
Need to get 2747 kB of archives.  
After this operation, 13.7 MB of additional disk space will be used.  
Do you want to continue? [Y/n]
```

Y mostramos por último la información del sistema Ubuntu Server en el que estamos.



1.3. Ejecución de Benchmarks en los S.O

He elegido los siguientes benchmarks:

- **Sudokut**: test escrito en Tcl (Tool command language) que mide el tiempo que tarda el procesador en resolver 100 Sudokus.
 - **Git**: test que mide el tiempo necesario para realizar algunas operaciones con Git.

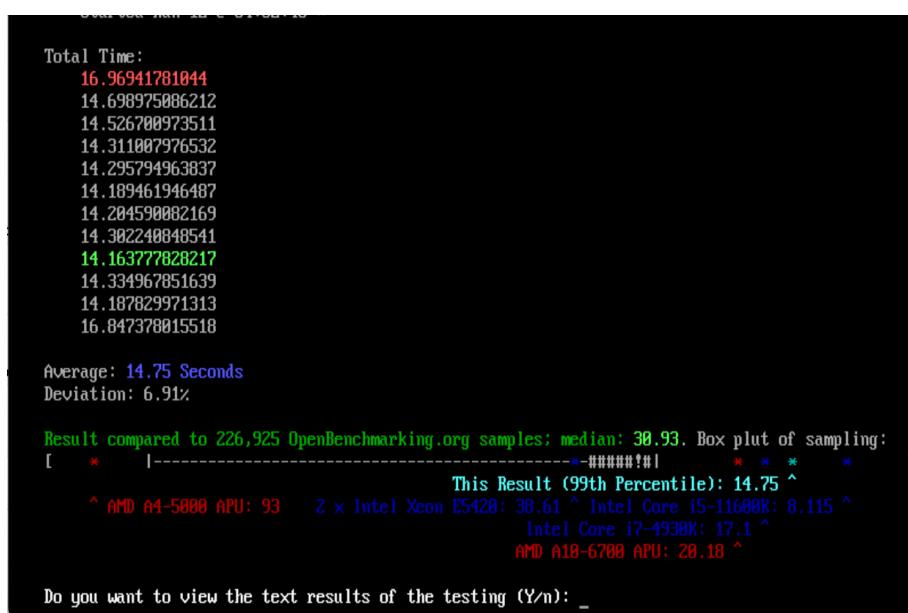
Como introducción, antes de mostrar las ejecuciones en los respectivos S.Os, es importante remarcar que Phoronix muestra la información del sistema antes de ejecutar cualquier benchmark además de instalar las dependencias necesarias para la ejecución de éste. Más tarde nos pide introducir un nombre para identificar a dicho benchmark y si deseamos, podemos escribir una descripción. Tras acabar de ejecutar el benchmark, nos pregunta si queremos ver los resultados y si queremos verlo en el sitio web de openbenchmarking.org.

1.3.1. Ejecución del benchmark Sudokut

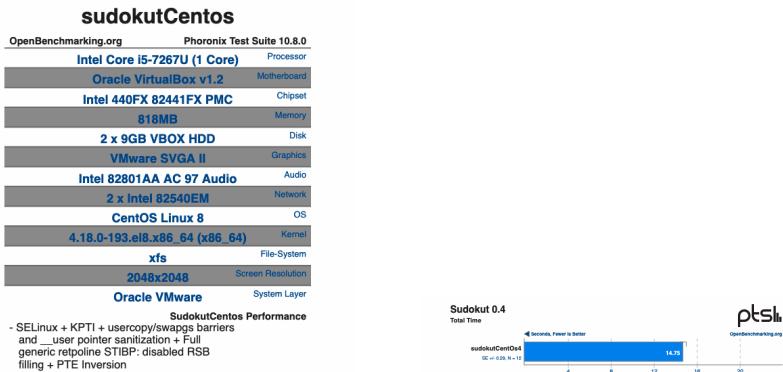
Para ejecutar dicho benchmark en ambos S.Os, utilizamos el siguiente comando,

```
1 $ phoronix-test-suite benchmark sudokut
```

Ejecución en CentOS



El resultado de dicha ejecución puede visualizarse aquí (<https://openbenchmarking.org/result/2201040-SP-SUDOKUTCE91>). Al ser la primera ejecución de un benchmark de Phoronix, mostraré los resultados que aparecen en dicho sitio web. Para futuros benchmarks, únicamente proporcionaré el acceso al resultado de la ejecución.



(a) SystemInfo

(b) Resultado del test

Ejecución en Ubuntu Server

El resultado de dicha ejecución puede visualizarse aqui (<https://openbenchmarking.org/result/2201042-IB-UBUNTUSUD19>).

Como vemos, ambas máquinas tardan aproximadamente lo mismo en ejecutar dicho benchmark. Sin embargo, si analizamos una a una las ejecuciones, vemos que, en general, tardamos un segundo menos en Ubuntu que en CentOS por lo que la máquina Ubuntu es más rápida.

1.3.2. Ejecución del benchmark Git

Para ejecutar dicho benchmark, utilizamos el siguiente comando,

```
1 $ phoronix-test-suite benchmark git
```

Ejecución en CentOS

```
CentOSExamenP2 [Running]
Running Interim Test Script @ 04:33:35
Started Run 2 @ 04:33:44
Running Interim Test Script @ 04:35:44
Started Run 3 @ 04:35:57
Running Interim Test Script @ 04:37:41
Started Run 4 @ 04:37:45 *
Running Interim Test Script @ 04:39:07
Started Run 5 @ 04:39:10 *
Running Interim Test Script @ 04:40:33
Started Run 6 @ 04:40:36 *
Running Interim Test Script @ 04:41:56
Started Run 7 @ 04:42:00 *
Running Interim Test Script @ 04:43:23
Started Run 8 @ 04:43:29 *
Running Interim Test Script @ 04:44:48
Started Run 9 @ 04:44:52 *
Running Post-Test Script @ 04:46:11

Time To Complete Common Git Commands:
114.7277104378
119.33715581894
102.01418696891
81.103268861771
81.580700874329
78.39213681221
82.431286811829
78.432648181915
78.520499126434

Average: 90.73 Seconds
Deviation: 18.36% 

Seconds < Lower Is Better
gitCentos4 ..... 90.73 I=====
Intel Core i5-7267U - VMware SUGA II - Oracle . 88.65 I=====

Do you want to view the text results of the testing (Y/n): _
```

El resultado de dicha ejecución puede visualizarse aquí (<https://openbenchmarking.org/result/2201049-SP-GITCENTOS89>).

Ejecución en Ubuntu Server

```
UbuntuServerP4 [Running]
under test.
Press ENTER to proceed without changes.

Current Description: Oracle VMware testing on Ubuntu 20.04 via the Phoronix Test Suite.

New Description:

[Performance Tip] The powersave CPU scaling governor is currently in use. It's possible to obtain greater performance if using the performance governor.

To change behavior, run:
echo performance | tee /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor

Reference: https://openbenchmarking.org/result/1706268-TR-CPUGOVERN32

To stop showing performance tips, run: phoronix-test-suite unload-module perf_tips

Continuing in 5 seconds or press CTRL-C to stop the testing process.

There is not enough space (at /home/alberto/.phoronix-test-suite/installed-tests/pts/git-1.1.0/) for this test to run.
```

Como vemos, nos da un error diciendo que no hay espacio para ejecutar dicho benchmark.

Tras varias peleas con el software y una reinstalación de Ubuntu Server de nuevo, con el doble de espacio que la máquina CentOS donde sí me dejó ejecutarlo, decidí realizar otro benchmark distinto en Ubuntu Server.

Systemd Total Boot Time Este benchmark usa la herramienta systemd-analyze para analizar el tiempo de arranque. De las opciones que nos ofrece dicho benchmark decidí ejecutar el test sobre el kernel.

```
UbuntuServerP4 [Running]
Started Run 1 @ 10:09:57
Test: Kernel:
14765
Average: 14765 ms

Comparison to 1,123 OpenBenchmarking.org samples since 17 August 2016; median result: 2220. Box
plot of samples:
[ *-----|-----*-----#####
This Result: 14765
Intel Xeon E5-1680 v3: 6617 ^ Intel Xeon E3-1260L v5: 700 ^
Intel Xeon E5-2609 v4: 1132 ^
Intel Core i3-7100: 1404 ^
Intel Core i9-9900KS: 1941 ^

Do you want to view the text results of the testing (Y/n): Y
ubuntuSystemD
Oracle VMWare testing on Ubuntu 20.04 via the Phoronix Test Suite.

Intel Core i5-7267U - VMware SVGA II - Oracle:
Processor: Intel Core i5-7267U (1 Core), Motherboard: Oracle VirtualBox v1.2, Chipset: Intel
440FX 82441FX PMC, Memory: 1024MB, Disk: 2 x 21GB VBOX HDD, Graphics: VMware SVGA II, Audio: Intel
H2801AA AC 97 Audio, Network: 2 x Intel 82540EM

OS: Ubuntu 20.04, Kernel: 5.4.0-91-generic (x86_64), Vulkan: 1.0.2, Compiler: GCC 9.3.0, Fil
e-System: ext4, Screen Resolution: 2048x2048, System Layer: Oracle VMWare

Systemd Total Boot Time
Test: Kernel
ms < Lower Is Better
Intel Core i5-7267U - VMware SVGA II - Oracle . 14765 |=====
Would you like to upload the results to OpenBenchmarking.org (y/n):
```

Los resultados de dicha ejecución pueden visualizarse aquí (<https://openbenchmarking.org/result/2201045-IB-UBUNTUSYS91>). Como vemos el benchmark ejecutado en Cen-

tOS es muy pesado y largo respecto a los vistos anteriormente. Estuvo aproximadamente 15 minutos ejecutándose y su tiempo medio fue de 90 segundos, siendo los resultados de las ejecuciones muy dispersos.

En Ubuntu Server, el resultado del benchmark es el esperado. Dicho tiempo de inicio (boot time) del kernel, debe ser muy pequeño para poder iniciar rápidamente el sistema.

2. Ejercicio JMeter

Tras probar un test básico para una web, utilizaremos Jmeter para hacer un test sobre una aplicación que ejecuta sobre dos contenedores (uno para la BD y otro para la aplicación en sí). El código está disponible en <https://github.com/davidPalomar-ugr/iseP4JMeter> donde se dan detalles sobre cómo ejecutar la aplicación en una de nuestras máquinas virtuales. El test de Jmeter debe incluir los siguientes elementos:

- El test debe tener parametrizados el Host y el Puerto en el Test Plan (puede hacer referencia usando \$param)

- Debe hacer dos grupos de hebras distintos para simular el acceso de los alumnos y los administradores. Las credenciales de alumno y administrador se cogen de los archivos: alumnos.csv y administrador.csv respectivamente.
- Añadimos esperas aleatorias a cada grupo de hebras (Gaussian Random Timer)
- El login de alumno, su consulta de datos (recuperar datos alumno) y login del administrador son peticiones HTTP.
- El muestreo para simular el acceso de los administradores lo debe coger el archivo apiAlumnos.log (usando un Acces Log Sampler).
- Use una expresión regular (Regular Expression Extractor) para extraer el token JWT que hay que añadir a la cabecera de las peticiones (usando HTTP Header Manager)

2.1. Instalación de Docker en Ubuntu Server

Realizamos los pasos del guión para instalar docker.

```

1 $ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
   add -
2 $ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/
   linux/ubuntu $(lsb_release -cs) stable"
3 $ sudo apt update

```

```

UbuntuServerP4 [Running]
albertolg@ubuntu:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
OK
albertolg@ubuntu:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
Hit:1 http://es.archive.ubuntu.com/ubuntu focal InRelease
Get:2 https://download.docker.com/linux/ubuntu focal InRelease [57.7 kB]
Get:3 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [13.5 kB]
Hit:5 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:6 http://es.archive.ubuntu.com/ubuntu focal-security InRelease
Get:7 https://download.docker.com/linux/ubuntu focal/stable amd64 Contents (deb) [1343 B]
Get:8 http://es.archive.ubuntu.com/ubuntu focal amd64 Contents (deb) [40.9 MB]
Get:9 http://es.archive.ubuntu.com/ubuntu focal-updates amd64 Contents (deb) [80.2 MB]
Get:10 http://es.archive.ubuntu.com/ubuntu focal-backports amd64 Contents (deb) [735 kB]
Get:11 http://es.archive.ubuntu.com/ubuntu focal-security amd64 Contents (deb) [71.3 MB]
Fetched 193 MB in 46s (4246 kB/s)
Reading package lists... Done
albertolg@ubuntu:~$ sudo apt update
Hit:1 https://download.docker.com/linux/ubuntu focal InRelease
Hit:2 http://es.archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Hit:4 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:5 http://es.archive.ubuntu.com/ubuntu focal-security InRelease
Fetched 114 kB in 1s (154 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
128 packages can be upgraded. Run 'apt list --upgradable' to see them.
albertolg@ubuntu:~$ _

```

Buscamos el repositorio de docker-ce (community edition)

```
1 $ apt search docker-ce
2 $ sudo apt install docker-ce
```

```
albertolg@ubuntu:~$ apt search docker-ce
Sorting... Done
Full Text Search... Done
docker-ce/focal 5:20.10.12~3-0~ubuntu-focal amd64
  Docker: the open-source application container engine

docker-ce-cli/focal 5:20.10.12~3-0~ubuntu-focal amd64
  Docker CLI: the open-source application container engine

docker-ce-rootless-extras/focal 5:20.10.12~3-0~ubuntu-focal amd64
  Rootless support for Docker.

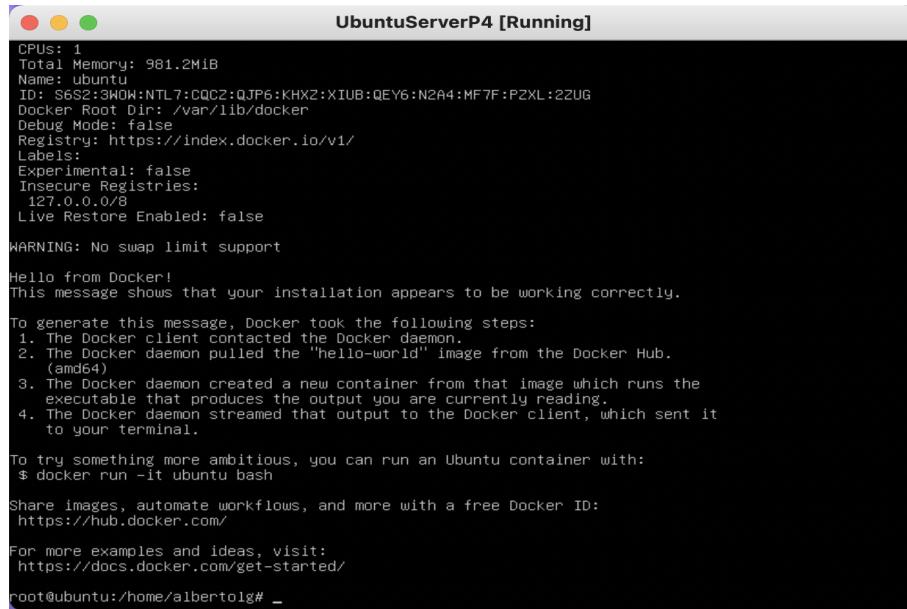
albertolg@ubuntu:~$ sudo apt install docker-ce
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-ce-cli docker-ce-rootless-extras docker-scan-plugin pigz slirp4netns
Suggested packages:
  aufs-tools cgroups-mount | cgroup-lite
The following NEW packages will be installed:
  containerd.io docker-ce docker-ce-cli docker-ce-rootless-extras docker-scan-plugin pigz
  slirp4netns
0 upgraded, 7 newly installed, 0 to remove and 128 not upgraded.
Need to get 97.2 MB of archives.
After this operation, 409 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Comprobamos e iniciamos el servicio con systemctl.

```
albertolg@ubuntu:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
  Active: inactive (dead)
TriggeredBy: ● docker.socket
  Docs: https://docs.docker.com
albertolg@ubuntu:~$ sudo systemctl start docker
albertolg@ubuntu:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2022-01-04 10:24:16 UTC; 2s ago
TriggeredBy: ● docker.socket
  Docs: https://docs.docker.com
    Main PID: 87778 (dockerd)
      Tasks: 7
     Memory: 40.7M
       CGroup: /system.slice/docker.service
               └─87778 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Jan 04 10:24:10 ubuntu dockerd[87778]: time="2022-01-04T10:24:10.479816744Z" level=warning msg="You>
Jan 04 10:24:10 ubuntu dockerd[87778]: time="2022-01-04T10:24:10.479929974Z" level=warning msg="You>
Jan 04 10:24:10 ubuntu dockerd[87778]: time="2022-01-04T10:24:10.480026147Z" level=warning msg="You>
Jan 04 10:24:10 ubuntu dockerd[87778]: time="2022-01-04T10:24:10.480256708Z" level=info msg="Loadin>
Jan 04 10:24:11 ubuntu dockerd[87778]: time="2022-01-04T10:24:11.238434987Z" level=info msg="Defaul>
Jan 04 10:24:12 ubuntu dockerd[87778]: time="2022-01-04T10:24:12.055584865Z" level=info msg="Loadin>
Jan 04 10:24:16 ubuntu dockerd[87778]: time="2022-01-04T10:24:15.785359109Z" level=info msg="Docker>
Jan 04 10:24:16 ubuntu dockerd[87778]: time="2022-01-04T10:24:15.785453783Z" level=info msg="Daemon>
Jan 04 10:24:16 ubuntu systemd[1]: Started Docker Application Container Engine.
Jan 04 10:24:16 ubuntu dockerd[87778]: time="2022-01-04T10:24:16.355577185Z" level=info msg="API li>
Lines 1-21/21 (END)
```

Añadimos nuestro usuario al grupo docker y probamos docker con hello-world.



```
CPU: 1  
Total Memory: 981.2MiB  
Name: ubuntu  
ID: S6S2:3HOW:NTL7:CQCZ:QJP6:KHXZ:XIUB:QEY6:N2A4:MF7F:PZXL:2ZUG  
Docker Root Dir: /var/lib/docker  
Debug Mode: false  
Registry: https://index.docker.io/v1/  
Labels:  
Experimental: false  
Insecure Registries:  
 127.0.0.0/8  
Live Restore Enabled: false  
  
WARNING: No swap limit support  
  
Hello from Docker!  
This message shows that your installation appears to be working correctly.  
  
To generate this message, Docker took the following steps:  
1. The Docker client contacted the Docker daemon.  
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.  
   (amd64)  
3. The Docker daemon created a new container from that image which runs the  
   executable that produces the output you are currently reading.  
4. The Docker daemon streamed that output to the Docker client, which sent it  
   to your terminal.  
  
To try something more ambitious, you can run an Ubuntu container with:  
$ docker run -it ubuntu bash  
  
Share images, automate workflows, and more with a free Docker ID:  
https://hub.docker.com/  
  
For more examples and ideas, visit:  
https://docs.docker.com/get-started/  
root@ubuntu:/home/alberto1g#
```

Instalamos ahora docker-compose

```
1 $ sudo apt install docker-compose  
2 $ docker-compose --version
```

```
root@ubuntu:/home/alberto1g# docker-compose --version  
docker-compose version 1.25.0, build unknown  
root@ubuntu:/home/alberto1g#
```

2.2. Instalación de la aplicación para el test con JMeter

Clonamos el repositorio del profesor David Palomar desde nuestra máquina Ubuntu.

```
root@ubuntu:/home/alberto1g# git clone https://github.com/davidPalomar-ugr/iseP4JMeter.git  
Cloning into 'iseP4JMeter'...  
remote: Enumerating objects: 3797, done.  
remote: Counting objects: 100% (23/23), done.  
remote: Compressing objects: 100% (14/14), done.  
remote: Total 3797 (delta 9), reused 15 (delta 7), pack-reused 3774  
Receiving objects: 100% (3797/3797), 7.79 MiB | 6.23 MiB/s, done.  
Resolving deltas: 100% (715/715), done.
```

Levantamos la aplicación con docker compose

```

root@ubuntu:/home/albertolg# cd iseP4JMeter/
root@ubuntu:/home/albertolg/iseP4JMeter# docker-compose up
Creating network "isep4jmeter_default" with the default driver
Pulling mongodb (mongo:)... .
latest: Pulling from library/mongo
7b1a6ab2e44d: Extracting [=====] 21.82MB/28.57MB
90eb44ebc60b: Download complete
5085b59f2efb: Download complete
c7499923d022: Download complete
019496b6c44a: Download complete
c0df4f407f69: Download complete
351daa315b6c: Download complete
a233e6240acc: Downloading [=====] 60.56MB/210MB
a3f57d6be64f: Download complete
dd1b5b345323: Download complete

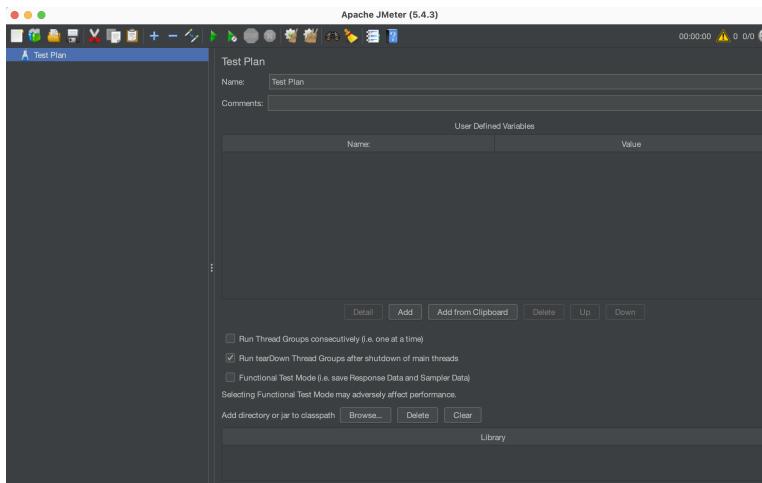
```

2.3. Instalación de JMeter en Host

Para instalar JMeter, simplemente nos vamos a la documentación de JMeter y nos descargamos el .zip desde su página de descargas y lo abrimos.

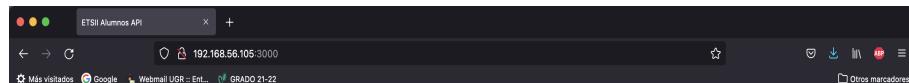
```
$ open apache-jmeter-5.4.3/bin/jmeter
```

The screenshot shows the Apache JMeter 5.4.3 download page. The page has a sidebar with links to Get Started, User Manual, Best Practices, Component Reference, Functions Reference, JavaDoc, and FAQ. Below the sidebar are sections for Tutorials, Community, and Foundation. The main content area is titled "Apache JMeter 5.4.3 (Requires Java 8+)" and contains sections for "Binaries" and "Source". Under "Binaries", there are two red-bordered download links: "apache-jmeter-5.4.3.tgz sha512.pgp" and "apache-jmeter-5.4.3.zip sha512.pgp". Under "Source", there are two download links: "apache-jmeter-5.4.3_src.tgz sha512.pgp" and "apache-jmeter-5.4.3_src.zip sha512.pgp". At the bottom, there is a link to "Browse download area".



2.4. Configuración del test con JMeter

Una vez levantada la aplicación clonada del git, podemos acceder en el navegador en: <http://192.168.56.105:3000>.



Parametrización del Host y el Puerto.

A screenshot of the JMeter Test Plan configuration interface. The "Test Plan" section shows "Name: TestPlanISEJMeterAlbertoLG" and "Comments: Test para Ejercicio 2 P4". Below this is a "User Defined Variables" table with two rows:

Name:	Value
HOST	192.168.56.105
PORT	3000

Creación de los grupos de hebras. Add - Threads(Users) - Thread Group y añadimos los grupos de hebras respectivos para los alumnos y para los administradores.

(a) Alumnos

(b) Administradores

Petición de acceso al servidor y Autorización a la API. Añadimos ahora la petición de acceso al servidor y la autorización básica a la API como se muestra en el repositorio git.

Creamos la petición HTTP: *Add - Config Element - HTTP Request Defaults*

Name:	Value	URL Encode?	Content-Type	Include Equals?
Name:	Value	URL Encode?	Content-Type	Include Equals?

Configuramos la autorización HTTP: *Add - Config Element - HTTP Authorization Manager*

Base URL	Username:	Password:	Domain	Realm	Mechanism
http://\${HOST}:\${PORT}	etsiiApi	*****			BASIC

Definimos ahora las credenciales para los alumnos especificando el fichero de donde queremos cogerlas, alumnos.csv. *Add - Config Element - CSV Data Set Config*

Creamos petición POST para los alumnos. *Add - Sampler - HTTP Request*

Send Parameters With the Request:				
Name:	Value	URL Encode?	Content-Type	Include Equals?
login	\${login}	<input checked="" type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>
password	\${password}	<input checked="" type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>

Tenemos que hacer que el test nos devuelva un OK además de un token tras la correcta autenticación en el servidor. *Add - Post Processors - Regular Expression Extractor*

Regular Expression Extractor

Name: Extract JWT Token

Comments:

Apply to:

Main sample and sub-samples Main sample only Sub-samples only JMeter Variable Name to use

Field to check:

Body Body (unesaped) Body as a Document Response Headers Request Headers URL Response Code

Name of created variable: token

Regular Expression: .+

Template (\$i\$ where i is capturing group number, starts at 1): \$0\$

Match No. (0 for Random):

Default Value:

Use empty default value

Esperas aleatorias a los grupos de hebras

Para que sea más realista añadimos esperas aleatorias en las peticiones donde éstas siguen una distribución Gaussiana. *Add - Timer - Gaussian Random Timer*

Gaussian Random Timer

Name: Espera un poco

Comments:

Thread Delay Properties

Deviation (in milliseconds): 100.0

Constant Delay Offset (in milliseconds): 300

Recuperación de los datos de los alumnos. Add - Sampler - HTTP Request

HTTP Request

Name: Recuperar Datos Alumno

Comments:

Basic Advanced

Web Server

Protocol [http]: Server Name or IP: Port Number:

HTTP Request

Method: GET Path: /api/v1/alumnos/alumno/\${__urlencode(\${login})} Content encoding:

Redirect Automatically Follow Redirects Use KeepAlive Use multipart/form-data Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	URL Encode?	Content-Type	Include Equals?
login	\${login}	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>

Detail Add Add from Clipboard Delete Up Down

Header por si falla el token. Add - Config Element - HTTP Header Manager

HTTP Header Manager

Name: JWT Token

Comments:

Headers Stored in the Header Manager

Name:	Value
Authorization	Bearer \${token}

Realizamos ahora los mismos pasos para los Administradores, pero en este caso cambiamos el archivo .csv por el correspondiente.

HTTP Request

Name: Login Administradores

Comments:

Basic Advanced

Web Server

Protocol [http]: Server Name or IP: Port Number:

HTTP Request

Method: POST Path: /api/v1/auth/login Content encoding:

Redirect Automatically Follow Redirects Use KeepAlive Use multipart/form-data Browser-compatible headers

Parameters Body Data Files Upload

Send Parameters With the Request:

Name:	Value	URL Encode?	Content-Type	Include Equals?
login	\${login}	<input checked="" type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>
password	\${password}	<input checked="" type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>

CSV Data Set Config

Name: Credenciales Administradores

Comments:

Configure the CSV Data Source

Filename: /iseP4Jmeter/JMeter/administradores.csv Browse...

File encoding:

Variable Names (comma-delimited): login,password

Ignore first line (only used if Variable Names is not empty): True

Delimiter (use '\t' for tab): ,

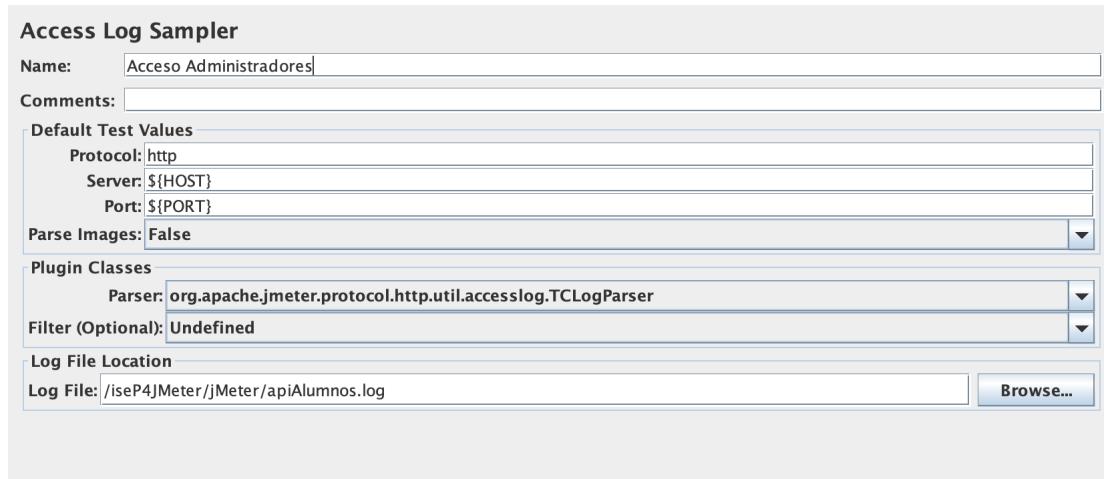
Allow quoted data?: False

Recycle on EOF ?: True

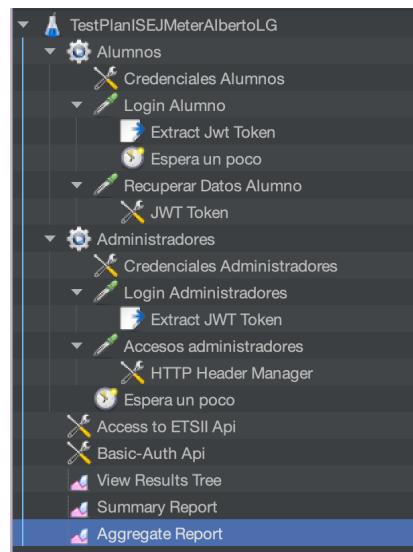
Stop thread on EOF ?: False

Sharing mode: All threads

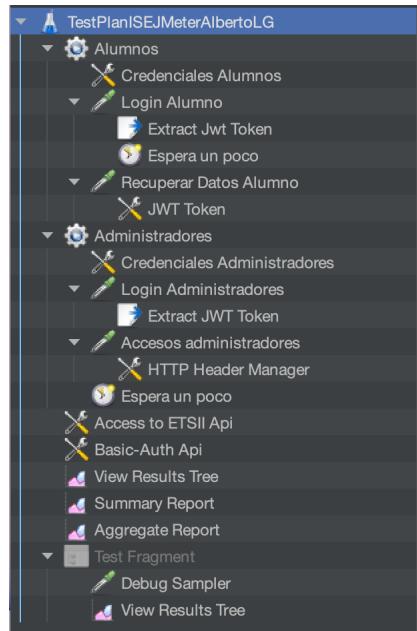
Los Administradores pueden ver quién entra por lo que mostramos el log de accesos. *Add - Sampler - Access Log Sampler*



Añadimos también los generadores de informes que se encuentran en *Add - Listener*



Por lo que el esquema del test quedaría de la siguiente forma tras añadir el fragmento de prueba *Add - Test Fragment*:



Si le damos a Play (triángulo verde en barra superior) podemos ver que se ejecuta correctamente el test y que sigue el esquema de la práctica.

Referencias

- OpenBenchmarking: <https://openbenchmarking.org>
- Phoronix Test Suite: <https://phoronix-test-suite.com/downloads>
- Documento Práctica 4 ISE
- GitHub David Palomar: <https://github.com/davidPalomar-ugr/iseP4JMeter>
- User Manual JMeter: <https://jmeter.apache.org/usermanual/index.html>