

# ISE P4 - Benchmarking y Ajuste del Sistema

enero del 2022

# Ejercicio 1 - Una vez haya indagado sobre los benchmarks disponibles, seleccione como mínimo dos de ellos y proceda a ejecutarlos en Ubuntu y CentOS. Comente las diferencias

Intentamos instalar phoronix en CentOS ejecutando `sudo yum install phoronix`. Nos dice que no encuentra ningún paquete llamado así, probamos con `sudo yum provides phoronix` y tampoco. Viendo la página web [openbenchmarking.org](http://openbenchmarking.org) vemos que para ejecutar un test se utiliza el comando `phoronix-test-suite benchmark [nombre del benchmark]`, así que intentamos instalar el paquete `phoronix-test-suite`. Tampoco lo conseguimos.

## Instalación en Ubuntu

Descargamos el paquete con `wget https://phoronix-test-suite.com/releases/repo/pts.debian/files/phoronix-test-suite_10.8.0_all.deb`. Lo instalamos con `sudo apt install ./phoronix-test-suite_10.8.0_all.deb`. Probamos a ejecutar un test sencillo con `phoronix-test-suite benchmark compress-7zip`. Esperamos a que se instalen las dependencias y esperamos a los resultados:

## Instalación en CentOS

Descargamos el código fuente con `curl https://phoronix-test-suite.com/releases/phoronix-test-suite-10.8.0.tar.gz > phoronix-test-suite-10.8.0.tar.gz`. Descomprimimos con `tar -xf phoronix-test-suite-10.8.0.tar.gz` y nos desplazamos a la carpeta que acabamos de descomprimir. Vemos que dentro de esta carpeta tenemos un ejecutable mediante `file phoronix-test-suite`. Podríamos ejecutarlo directamente, pero decidimos instalarlo ejecutando `install-sh`.

Intentamos ejecutar `phoronix-test-suite benchmark compress-7zip` para comprobar que funciona, pero nos pide instalar php. Lo instalamos. Nos pide las siguientes extensiones de PHP:

Instalamos los paquetes con `sudo yum install php-dom php-json`. Intentamos ejecutar de nuevo el test y nos indica que no ha sido capaz de instalar una de las dependencias, 7zip. La instalamos manualmente con `sudo yum install epel-release.noarch && sudo yum install p7zip`.

Intentamos ejecutar de nuevo el benchmark de prueba. Esta vez tenemos éxito.

Para el benchmark en sí obtenemos los siguientes resultados:

## Ejecución de dos benchmarks

Elegimos ejecutar los benchmarks `x264` y `fio` (Flexible IO tester).

En CentOS no consigue instalar correctamente `x264`. Resolvemos manualmente instalando el repositorio rpmfusion y acto seguido `x264`. En Ubuntu instalamos las dependencias sin problemas.

Tanto para CentOS como para Ubuntu necesitamos instalar el paquete `zlib-dev` para ejecutar el benchmark `fio`.

Observamos que los resultados son bastante parecidos entre ambas máquinas, aunque un tanto mejores en la máquina del Ubuntu. No sabemos exactamente a qué puede deberse (ambas máquinas están virtualizadas de la misma manera, con las mismas especificaciones y los benchmarks no se hicieron simultáneamente en ambas máquinas), ¿tal vez podría ser porque Ubuntu está más optimizado para ejecutarse en máquinas virtuales? ¿o tal vez porque CentOS al ser más conservador en el compilador y el kernel no dispone de un software mejor optimizado?

● ○ ● daniel — danielpm@danielpm-ubuntu-ise: ~ — ssh danielpm@ise.moemoe....

## System Information

```
:           AMD Ryzen 7 2700X Eight-Core
Core Count:      1
Extensions:     SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE
Cache Size:    16 MB
Microcode:     0x6000626
Core Family:   Zen+


:           VMware SVGA II
Vulkan:        1.0.2
Screen:         2048x2048


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


:           8GB

:           32GB VBOX HDD
File-System:   ext4
Mount Options: relatime rw
Disk Scheduler: MQ-DEADLINE
Disk Details:  Block Size: 4096


:           Ubuntu 20.04
Kernel:        5.4.0-91-generic (x86_64)

Compiler:       GCC 9.3.0

System Layer:  Oracle VMware

Security:      itlb_multihit: Not affected
                + l1tf: Not affected
                + mds: Not affected
                + meltdown: Not affected
                + spec_store_bypass: Mitigation of SSB disabled via pr
ctl and seccomp
                + spectre_v1: Mitigation of usercopy/swapgs barriers a
nd __user pointer sanitization
                + spectre_v2: Mitigation of Full AMD retrpoline STIBP:
disabled RSB filling
                + srbds: Not affected
                + tsx_async_abort: Not affected
```

Figure 1: *Phoronix System Information* de la máquina virtual ejecutando Ubuntu

```

daniel — danielpm@danieldpm-ubuntu-ise: ~ — ssh danielpm@ise.moemoe.... Would you like to save these test results (Y/n): [ ] 
Enter a name for the result file: [ ] 
Enter a name for the result file: t1 [ ] 
Enter a unique name to describe this test run / configuration: t1 [ ] 

If desired, enter a new description below to better describe this result set / system configuration under test. 
Press ENTER to proceed without changes. 

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

[N]ew Description: [ ] 

7-Zip Compression 21.06: pts/compress-7zip-1.8.0 
Test 1 of 1 
Estimated Trial Run Count: 3 
Estimated Time To Completion: 4 Minutes [11:24 UTC] 
    Started Run 1 @ 11:20:56 
    Started Run 2 @ 11:21:34 
    Started Run 3 @ 11:22:11 

Test: Compression Rating: 
    4736 
    4737 
    4745 

Average: 4739 MIPS 
Deviation: 0.10% 

Comparison to 419 OpenBenchmarking.org samples since 25 November; median result: 52225. Box plot of samples: [-*#*#!#####-----*-----| ] 
    ^ Ryzen 7 4700U: 34892 2 x EPYC 75F3: 394245 ^ 
    ^ Xeon E3-1235L v5: 17239 

Test: Decompression Rating: 
    4064 
    4050 
    4036 

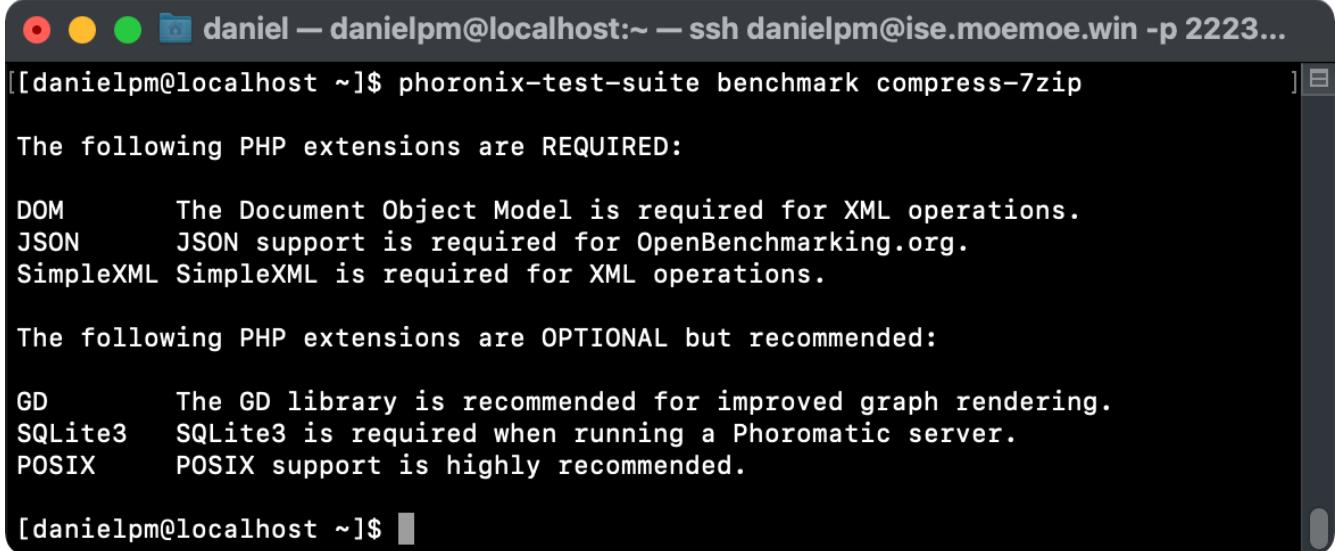
Average: 4050 MIPS 
Deviation: 0.35% 

Comparison to 417 OpenBenchmarking.org samples since 25 November; median result: 43979. Box plot of samples: [-#*!*#*-----*-----| ] 
    ^ Core i9-10980XE: 101750 Ampere ARMv8 Neoverse-N1: 668912 ^ 
    ^ Core i9-10900K: 71920 
    ^ Ryzen 7 3700X: 59865 
    ^ Core i5-10600K: 40676 

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

```

Figure 2: Resultados del benchmark de 7zip en Ubuntu



The screenshot shows a terminal window with the following content:

```
daniel — danielpm@localhost:~ — ssh danielpm@ise.moemoe.win -p 2223...
[[danielpm@localhost ~]$ phoronix-test-suite benchmark compress-7zip
The following PHP extensions are REQUIRED:
DOM      The Document Object Model is required for XML operations.
JSON     JSON support is required for OpenBenchmarking.org.
SimpleXML SimpleXML is required for XML operations.

The following PHP extensions are OPTIONAL but recommended:
GD        The GD library is recommended for improved graph rendering.
SQLite3   SQLite3 is required when running a Phoromatic server.
POSIX     POSIX support is highly recommended.

[danielpm@localhost ~]$ ]
```

Figure 3: Error de Phoronix en CentOS

● ○ ● daniel — danielpm@localhost:~ — ssh danielpm@ise.moemoe.win -p 2223...

## System Information

```
:           AMD Ryzen 7 2700X Eight-Core
Core Count:      1
Extensions:     SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE
Cache Size:     16 MB
Microcode:      0x6000626
Core Family:    Zen+


:           VMware SVGA II
Screen:        2048x2048


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


:           8GB

:           32GB VBOX HDD
File-System:   xfs
Mount Options: attr2 inode64 noquota relatime rw seclabel
Disk Scheduler: MQ-DEADLINE
Disk Details:  Block Size: 4096


:           CentOS Linux 8
Kernel:        4.18.0-193.el8.x86_64 (x86_64)

Compiler:      GCC 8.5.0 20210514

System Layer:  Oracle VMware

Security:      SELinux
               + itlb_multihit: Not affected
               + l1tf: Not affected
               + mds: Not affected
               + meltdown: Not affected
               + spec_store_bypass: Mitigation of SSB disabled via pr
ctl and seccomp
               + spectre_v1: Mitigation of usercopy/swapgs barriers a
nd __user pointer sanitization
               + spectre_v2: Mitigation of Full generic retpoline STI
BP: disabled RSB filling
               + tsx_async_abort: Not affected


Would you like to save these test results (Y/n): █
```

Figure 4: *Phoronix System Information* de la máquina virtual ejecutando CentOS

```

[Would you like to save these test results (Y/n): Y]
[Enter a name for the result file: t1]
[Enter a unique name to describe this test run / configuration: t1]

>If desired, enter a new description below to better describe this result set / system configuration under test.
>Press ENTER to proceed without changes.

Current Description: Oracle VMware testing on CentOS Linux 8 via the Phoronix Test Suite.

>New Description:

7-Zip Compression 21.06:
pts/compress-7zip-1.8.0
Test 1 of 1
Estimated Trial Run Count: 3
Estimated Time To Completion: 4 Minutes [07:15 EST]
    Started Run 1 @ 07:11:47
    Started Run 2 @ 07:12:24
    Started Run 3 @ 07:13:02

Test: Compression Rating:
4962
4822
4934

Average: 4906 MIPS
Deviation: 1.51%

Comparison to 419 OpenBenchmarking.org samples since 25 November; median result: 52225. Box plot of samples:
[-*#*#!#####-----*-----|]
    ^ Ryzen 7 4700U: 34892 2 x EPYC 75F3: 394245 ^
    ^ Xeon E3-1235L v5: 17239

Test: Decompression Rating:
3965
3991
3974

Average: 3977 MIPS
Deviation: 0.33%

Comparison to 417 OpenBenchmarking.org samples since 25 November; median result: 43979. Box plot of samples:
[-#*!*#*-----*-----|]
    ^ Core i9-10980XE: 101750 Ampere ARMv8 Neoverse-N1: 668912 ^
    ^ Core i9-10900K: 71920
    ^ Ryzen 7 3700X: 59865
    ^ Core i5-10600K: 40676

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

```

Figure 5: Resultados del benchmark de 7zip en CentOS

```

x264 2019-12-17:
pts/x264-2.6.1
Test 1 of 1
Estimated Trial Run Count: 3
Estimated Time To Completion: 4 Minutes [07:39 EST]
    Started Run 1 @ 07:36:07
    Started Run 2 @ 07:37:09
    Started Run 3 @ 07:38:10

H.264 Video Encoding:
10.4
10.44
10.42

Average: 10.42 Frames Per Second
Deviation: 0.19%

Comparison to 2,851 OpenBenchmarking.org samples since 26 February 2011; median result: 81.36. Box plot of samples:
[|*-----#####*! # #*#-----*-----*-*|* ]
  ^ This Result (9th Percentile): 10.42
    Core i9-9900K: 104 ^ EPYC 7502P: 168 ^ EPYC 75F3: 229 ^
    Ryzen 7 PRO 4750G: 99 ^
    EPYC 7F32: 93 ^
    Ryzen 7 4800H: 86 ^
      2 x EPYC 7713: 222 ^
      2 x EPYC 73F3: 215 ^
      2 x EPYC 7642: 209 ^

[danielpm@danielpm@localhost ~]$ 

x264 2019-12-17:
pts/x264-2.6.1
Test 1 of 1
Estimated Trial Run Count: 3
Estimated Time To Completion: 4 Minutes [12:34 UTC]
    Started Run 1 @ 12:31:32
    Started Run 2 @ 12:32:35
    Started Run 3 @ 12:33:39

H.264 Video Encoding:
10.23
10.67
10.35

Average: 10.22 Frames Per Second
Deviation: 1.37%

Comparison to 2,851 OpenBenchmarking.org samples since 26 February 2011; median result: 81.36. Box plot of samples:
[|*-----#####*! # #*#-----*-----*-*|* ]
  ^ This Result (9th Percentile): 10.22
    Core i9-9900K: 104 ^ EPYC 7502P: 168 ^ EPYC 75F3: 229 ^
    Ryzen 7 PRO 4750G: 99 ^
    EPYC 7F32: 93 ^
    Ryzen 7 4800H: 86 ^
      2 x EPYC 7713: 222 ^
      2 x EPYC 73F3: 215 ^
      2 x EPYC 7642: 209 ^

danielpm@danielpm-ubuntu-ise:$ 

```

Figure 6: Ejecución del benchmark x264 en ambas máquinas

```

daniel — danielpm@localhost:~ — ssh danielpm@ise.moemoe.win -p 2223...
Type: Random Read - Engine: Linux AIO - Buffered: Yes - Direct: No - Block Size: 4MB - Disk Target: Default Test Directory:
96.9
103
97.2
97.1
102
97
103
96
95.5
102
97.8
101
97.1
106
97

Average: 99.2 MB/s
Deviation: 3.27%
Samples: 15

Type: Random Read - Engine: Linux AIO - Buffered: Yes - Direct: No - Block Size: 4MB - Disk Target: Default Test Directory:
21
22
21
21
22
21
22
28
20
22
21
22
21
23
21

Average: 21 IOPS
Deviation: 3.83%
Samples: 15

daniel — danielpm@localhost:~ — ssh danielpm@ise.moemoe....@162.25.14.162:~$ 
Started Run 14 @ 16:31:22 *
Running Post-Test Script @ 16:32:01

Type: Random Read - Engine: Linux AIO - Buffered: Yes - Direct: No - Block Size: 4MB - Disk Target: Default Test Directory:
102
109
107
108
109
111
108
102
107
110
108
109
110
109

Average: 108 MB/s
Deviation: 2.50%
Samples: 14

Type: Random Read - Engine: Linux AIO - Buffered: Yes - Direct: No - Block Size: 4MB - Disk Target: Default Test Directory:
22
23
23
23
24
24
23
22
23
24
23
24
24
24

Average: 23 IOPS
Deviation: 3.12%
Samples: 14

danielpm@danielpm-ubuntu-ise:$ 

```

Figure 7: Ejecución de uno de los benchmark en fio en ambas máquinas

Ejercicio 1 opcional - Pruebe a ejecutar uno de los benchmarks de los seleccionados en el ejercicio anterior y analice las diferencias en los resultados obtenidos en la MV directamente.

Para descargar el contenedor nos vamos a la página de descargas de phoronix. Nos indican la página en el hub de docker del contenedor. Ahí vemos que podemos descargarlo ejecutando `docker pull phoronix/pts`. Lo ejecutamos en Ubuntu.

En la documentación de docker ([https://docs.docker.com/engine/reference/commandline/container\\_start/](https://docs.docker.com/engine/reference/commandline/container_start/)) vemos que para poner en marcha el repositorio tenemos que ejecutar `sudo docker run -it phoronix/pts`. Lo ejecutamos.

Figure 1: Login dentro del contenedor de *Phoronix*

Ejecutamos el test **fio**. No encontramos forma de instalar las dependencias. Ejecutamos el test PHPBench en momentos diferentes en el contenedor y en el host.

```

daniel — danielpm@danielpm-ubuntu-ise: ~ — ssh danielpm@ise.moemoe...
Estimated Trial Run Count: 3
Estimated Time To Completion: 4 Minutes [20:30 UTC]
Started Run 1 @ 20:26:16
Started Run 2 @ 20:26:58
Started Run 3 @ 20:27:39

PHP Benchmark Suite:
529696
532379
527313

Average: 529793 Score
Deviation: 0.48%

Comparison to 227,644 OpenBenchmarking.org samples since 26 February 2011; median result: 332117. Box plot of samples:
[ |*---#####-----|#####
This Result (95th Percentile): 529793 ^
^ Xeon E31230: 32882 Core i7-5600U: 476418 ^ Core i7-8559U: 703476 ^
Core i5-8500: 684763 ^
Xeon E3-1270 v5: 664034 ^
Xeon Gold 5220R: 636640 ^

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

```

daniel — danielpm@danielpm-ubuntu-ise: ~ — ssh danielpm@ise.moemoe...
Estimated Trial Run Count: 3
Estimated Time To Completion: 4 Minutes [20:32 UTC]
Started Run 1 @ 20:29:04
Started Run 2 @ 20:29:46
Started Run 3 @ 20:30:29

PHP Benchmark Suite:
529277
518865
526650

Average: 524664 Score
Deviation: 1.12%

Comparison to 227,644 OpenBenchmarking.org samples since 26 February 2011; median result: 332117. Box plot of samples:
[ |*---#####-----|#####
This Result (95th Percentile): 524664 ^
^ Xeon E31230: 32882 Core i7-5600U: 476418 ^ Core i7-8559U: 703476 ^
Core i5-8500: 684763 ^
Xeon E3-1270 v5: 664034 ^
Xeon Gold 5220R: 636640 ^

danielpm@danielpm-ubuntu-ise:-$
```

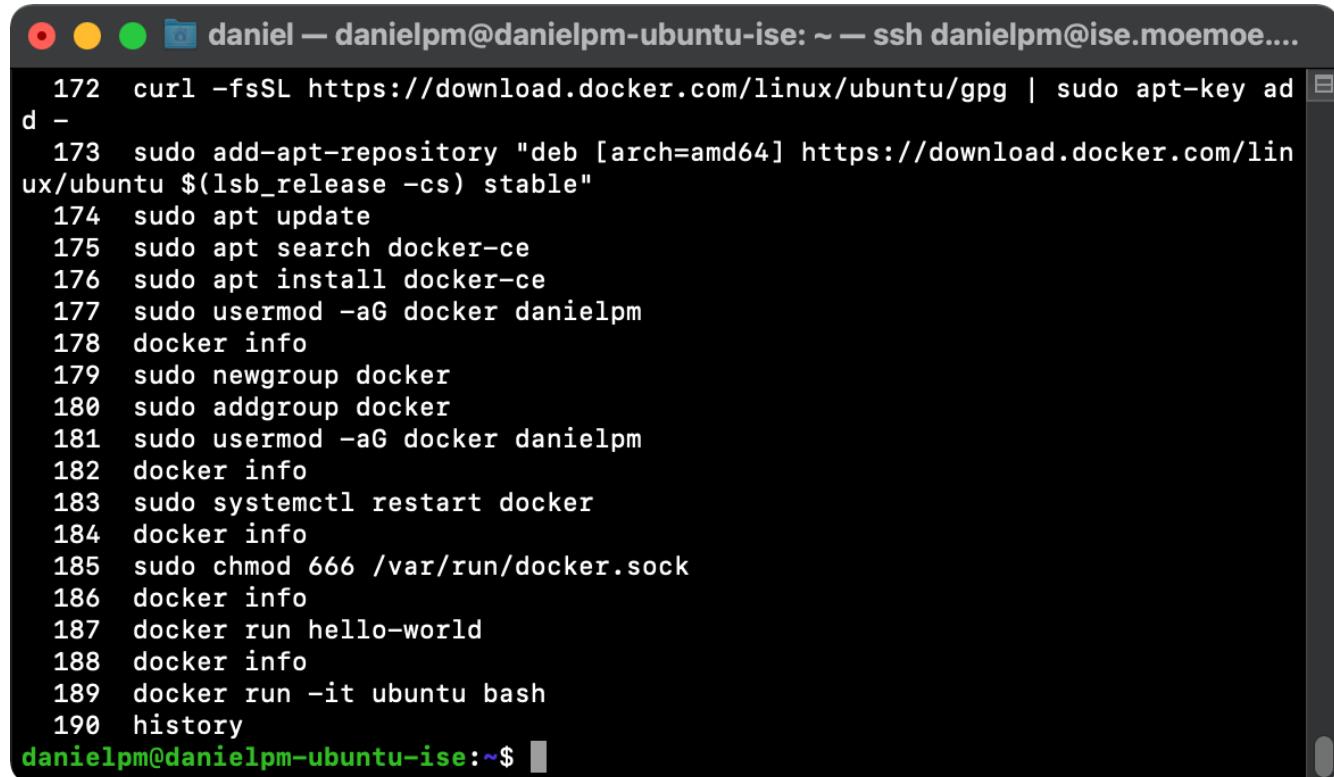
Figure 2: Comparación de benchmark en el contenedor (izquierda) y sobre el host (derecha)

En la figura 2 podemos observar cómo ambos resultados son prácticamente idénticos. La razón de esto es que docker (sobre Linux al ejecutar un contenedor Linux) no utiliza un hipervisor, sino que aisla los recursos de la máquina virtual ejecutándose directamente sobre el kernel de Linux del host (simplificado).

## Ejercicio 2 - Utilizaremos Jmeter para hacer un test sobre una aplicación que ejecuta sobre dos contenedores.

### Instalación de Docker

Primero instalamos docker en la máquina virtual de Ubuntu.



A terminal window titled "daniel — danielpm@danielpm-ubuntu-ise: ~ — ssh danielpm@ise.moemoe....". The window contains a series of terminal commands for installing Docker on an Ubuntu system. The commands are numbered from 172 to 190. The session starts with curling the Docker GPG key, adding the Docker repository, updating the package list, installing Docker CE, and modifying user permissions. It then checks Docker info, creates a new group for Docker, adds the user to the group, restarts the Docker service, changes file permissions on the socket, and runs a hello-world container. Finally, it runs an Ubuntu bash shell and lists the command history.

```
172 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
173 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
174 sudo apt update
175 sudo apt search docker-ce
176 sudo apt install docker-ce
177 sudo usermod -aG docker danielpm
178 docker info
179 sudo newgroup docker
180 sudo addgroup docker
181 sudo usermod -aG docker danielpm
182 docker info
183 sudo systemctl restart docker
184 docker info
185 sudo chmod 666 /var/run/docker.sock
186 docker info
187 docker run hello-world
188 docker info
189 docker run -it ubuntu bash
190 history
danielpm@danielpm-ubuntu-ise:~$
```

Figure 1: Comandos ejecutados para instalar docker

A la hora de ejecutar docker info nos encontramos con que docker nos indica que no tiene permisos para abrir un archivo socket. Después de intentar arreglarlo reiniciando el servicio, seguimos la solución dada en <https://newbedev.com/shell-error-got-permission-denied-while-trying-to-connect-to-the-docker-daemon-socket> (ejecutar sudo chmod 666 sobre el archivo para darle permisos)

Ejecutamos entonces dos contenedores de ejemplo, uno *hello world* y otro con Ubuntu. Comprobamos que funciona correctamente.

### Descarga e instalación del repositorio iseP4JMeter

Clonamos el repositorio que se nos indica en el ejercicio, nos desplazamos al directorio donde se ha clonado (~/iseP4JMeter) e intentamos ejecutar docker-compose up para poner en marcha la aplicación. Vemos que no está instalado, así que lo instalamos con sudo apt install docker-compose. Probamos de nuevo y comprobamos cómo se empiezan a descargar y configurar los componentes necesarios para formar los contenedores.

Comprobamos que funciona correctamente introduciendo en el navegador 192.168.0.140:3000

### 2.1 Parametrización del host y el puerto en el Test Plan

Dentro de JMeter, en el Test Plan que se nos crea por defecto, creamos dos variables, Host y Puerto.

```

daniel — danielpm@danielpm-ubuntu-ise: ~ — ssh danielpm@ise.moemoe.... at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/in fo": dial unix /var/run/docker.sock: connect: permission denied errors pretty printing info [danielpm@danielpm-ubuntu-ise:~$ sudo systemctl restart docker] [danielpm@danielpm-ubuntu-ise:~$ docker info]
Client:
Context: default
Debug Mode: false
Plugins:
app: Docker App (Docker Inc., v0.9.1-beta3)
buildx: Docker Buildx (Docker Inc., v0.7.1-docker)
scan: Docker Scan (Docker Inc., v0.12.0)

Server:
ERROR: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%2Fdocker.sock/v1.24/in fo": dial unix /var/run/docker.sock: connect: permission denied errors pretty printing info [danielpm@danielpm-ubuntu-ise:~$ sudo chmod 666 /var/run/docker.sock] [danielpm@danielpm-ubuntu-ise:~$ docker info]
Client:
Context: default

```

Figure 2: docker: connect: permission denied



### ETSII Alumnos API

```

Descripción de la API Restful:
POST /api/v1/auth/login
    Parámetros:
        login:<emailUsuario>
        password:<secreto>
    Seguridad:
        Acceso protegido con BasicAuth (etsiiApi:laApiDeLaETSIIDaLache)
    Retorna:
        JWT Token

GET /api/v1/alumnos/alumno/<email>
    Seguridad:
        Token JWT valido en cabecera estandar authorization: Bearer <token>
        Alumnos solo pueden solicitar sus datos. Administradores pueden solicitar cualquier alumno válido
    Retorna:
        Objeto Json con perfil de alumno

```

Figure 3: Descripción básica de la API *ETSII Alumnos*

```

echo "ERROR: Curl fallo con resultado: $resultado"
fi

curl \
-H "Authorization: Bearer $TOKEN" \
http://$SERVER:3000/api/v1/alumnos/alumno/mariweiss%40tropoli.com
[danielpm@danielpm-ubuntu-ise:~/iseP4JMeter$ ./pruebaEntorno.sh
{"_id":"61d71e12a337584fe7f50607","nombre":"Mari","apellidos":"Fletcher Weiss","sexo":"female","email":"mariweiss@tropoli.com","fechaNacimiento":"1992-04-04T00:00:00.000Z","comentarios":"Aliquip dolor laboris ullamco id ex labore. Ipsum eiusmod ut aliquip non cillum deserunt sunt commodo anim ad nisi excepteur eu deserunt. Sit sunt proident Lorem irure irure minim adipisicing cillum. Nostrud officia in proident velit velit fugiat pariatur quis ad laboris minim dolor elit. Sint velit pariatur commodo sint veniam exercitation. Duis proident minim consequat consectetur sint et tempor labore culpa esse. Exercitation laborum non esse mollit tempor ea dolor minim adipisicing mollit in a liqua.\r\nUllamco adipisicing excepteur commodo sunt nulla quis sunt velit Lorem pariatur sunt ad do incididunt. In eu nostrud ullamco laboris eu minim. Consequat sit et eiusmod officia ex sit minim sit laborum quis laborum labore non. Dolor nulla ut pariatur reprehenderit minim dolore consequat sunt aliquip ipsum esse. Excepteur consequat fugiat elit et nisi dolore aute minim nostrud et.\r\n","cursos":[{"_id":"61d729db851192e3becffbe","curso":1,"media":5.2},{"_id":"61d729db851192e3becffbf","curso":2,"media":9.1}],"usuario":10}
[danielpm@danielpm-ubuntu-ise:~/iseP4JMeter$ ]

```

Figure 4: Ejecución del script *pruebaEntorno.sh*

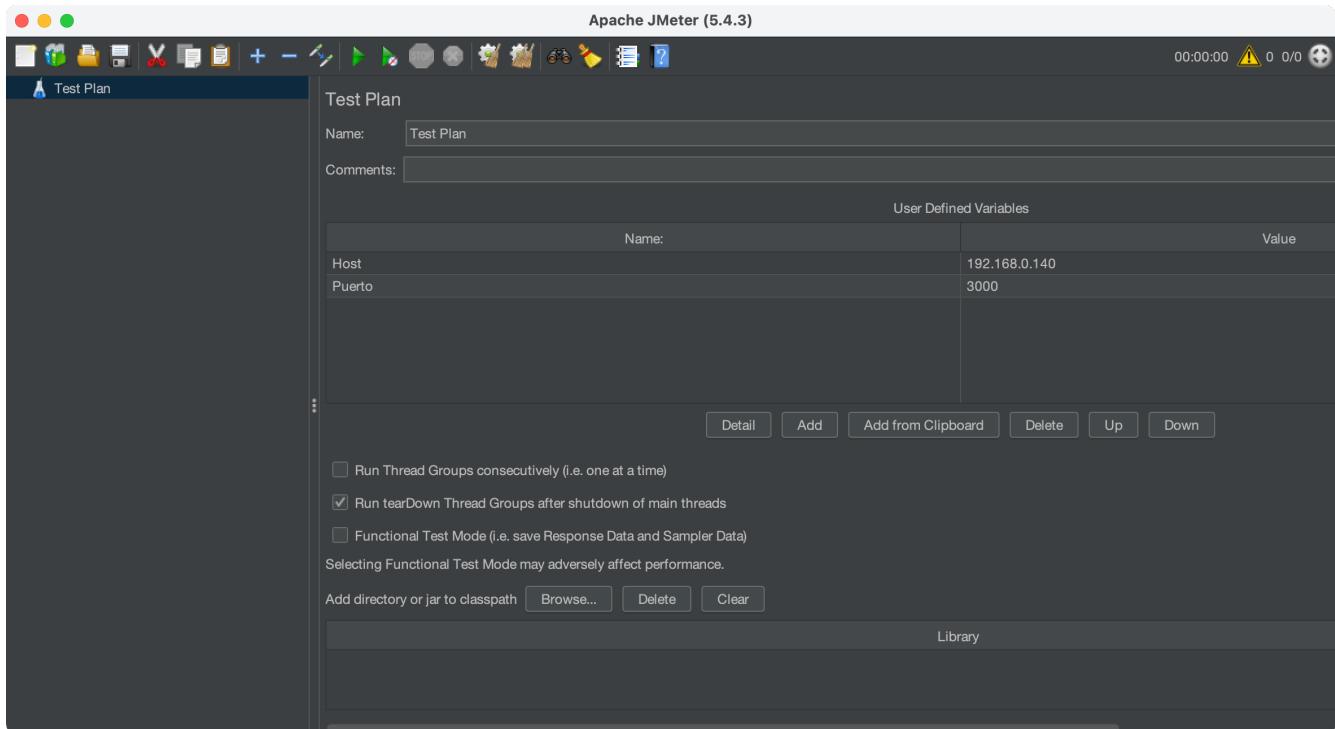


Figure 5: *User Defined Variables* donde vemos el Host y el Puerto

## 2.2 Creación de dos grupos de hebras distintos para simular el acceso de los alumnos y los administradores

Creamos dos Thread Group, uno para los alumnos y otro para los administradores.

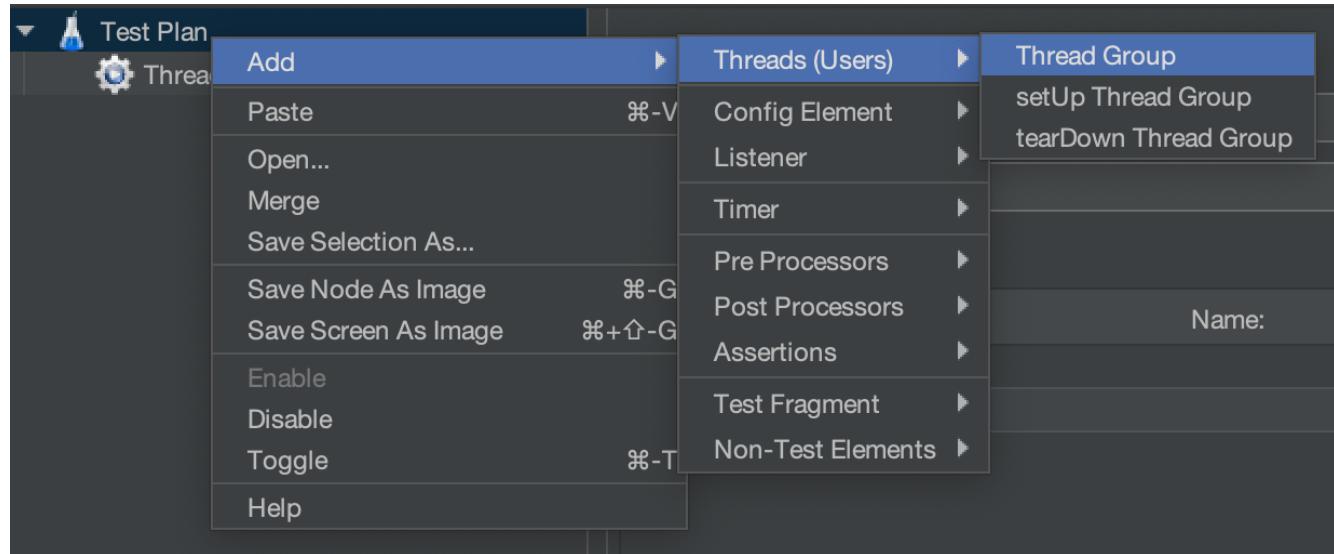


Figure 6: Menú para crear un *Thread Group*

Configuramos el Host y el Port en ambos. Vemos que en el archivo `pruebaEntorno.sh` las peticiones HTTP que se realizan son de tipo POST y sobre `/api/v1/....`

Extraemos la información sobre el login del archivo `alumnos.csv`

Name:	Value:	URL Encode?	Content-Type	Include Equals?
login	rowlandsaunders@tropoli.com	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>
password	exercitation	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>

Figure 7: Configuración (errónea) del HTTP POST Request del Alumno

Vemos en el *Results Tree* (src: <https://stackoverflow.com/questions/1515689/jmeter-how-to-log-the-full-request-for>) si funciona correctamente tras ejecutar alumnos -> Start. No funciona correctamente. Corregimos (src: <https://jmeter.apache.org/usermanual/functions.html>)

Vemos que nos vuelve a fallar (401: Unauthorized). Revisamos el script de nuevo y vemos que nos falta la contraseña y activar el URL ENCODE.

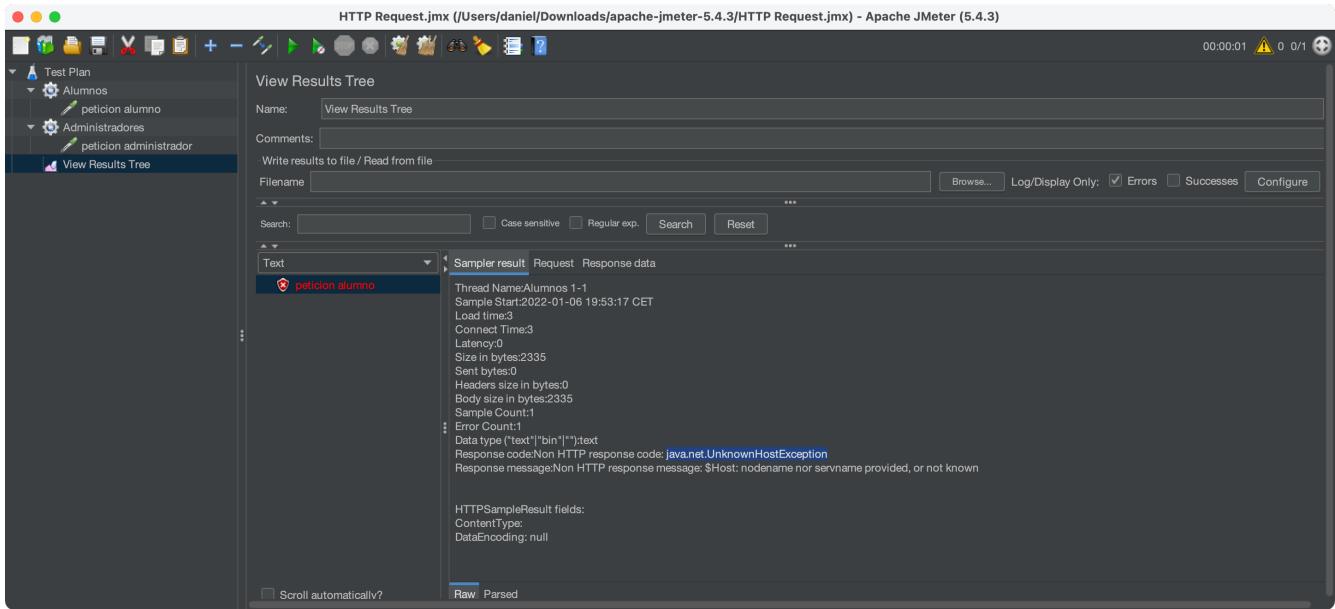


Figure 8: `java.net.UnknownHostException` error dado por la configuración errónea del HTTP POST Request del Alumno

Para la contraseña del *Basic Auth* vemos en <https://www.blazemeter.com/blog/how-use-http-basic-authentication-jmeter> que hay que crear un **HTTP Authorization Manager**.

Vemos que por fin tenemos acceso a la API.

Repetimos los pasos anteriores para el *Thread Group* de Administradores (esta vez de forma correcta) y de igual forma comprobamos que funciona correctamente.

## 2.3 Añadimos esperas aleatorias a cada grupo de hebras (Gaussian Random Timer)

Añadimos un Gaussian Random Timer desde Alumnos -> Add -> Timer

Comprobamos que funciona correctamente lanzando 10 hebras

## 2.4 El login de alumno, su consulta de datos y login del administrador son peticiones HTTP

Podemos comprobar en apartados anteriores que el login del alumno y del administrador son peticiones HTTP, aunque no hemos creado una para su consulta de datos. La creamos.

Vemos que en `http://192.168.0.140:3000` se nos indica que para recuperar la información de un alumno tenemos que hacer una petición de tipo GET sobre `/api/v1/alumnos/alumno/<email>`

Recibimos un error 401 (Unauthorized). Sospechamos que es porque necesitamos un token JWT válido para hacer la petición.

## 2.6 (Interludio) Use una expresión regular para extraer el token JWT que hay que añadir a la cabecera de las peticiones.

Nos vamos al View Results Tree para ver exactamente qué patrón tenemos que extraer de la respuesta del servidor. Abrimos una petición correcta cualquiera e inspeccionamos el response body y el response header. En el response header no identificamos nada que pueda servir como JWT, suponemos que el response body entero es el JWT en sí (src: <https://www.blazemeter.com/blog/using-regular-expressions-to-extract-tokens-and-session-ids-to-variables>). Comprobamos que funciona añadiendo `log.info("*****"+vars.get("XSRF-TOKEN"))` a un JSR233 PostProcessor. Vemos que no.

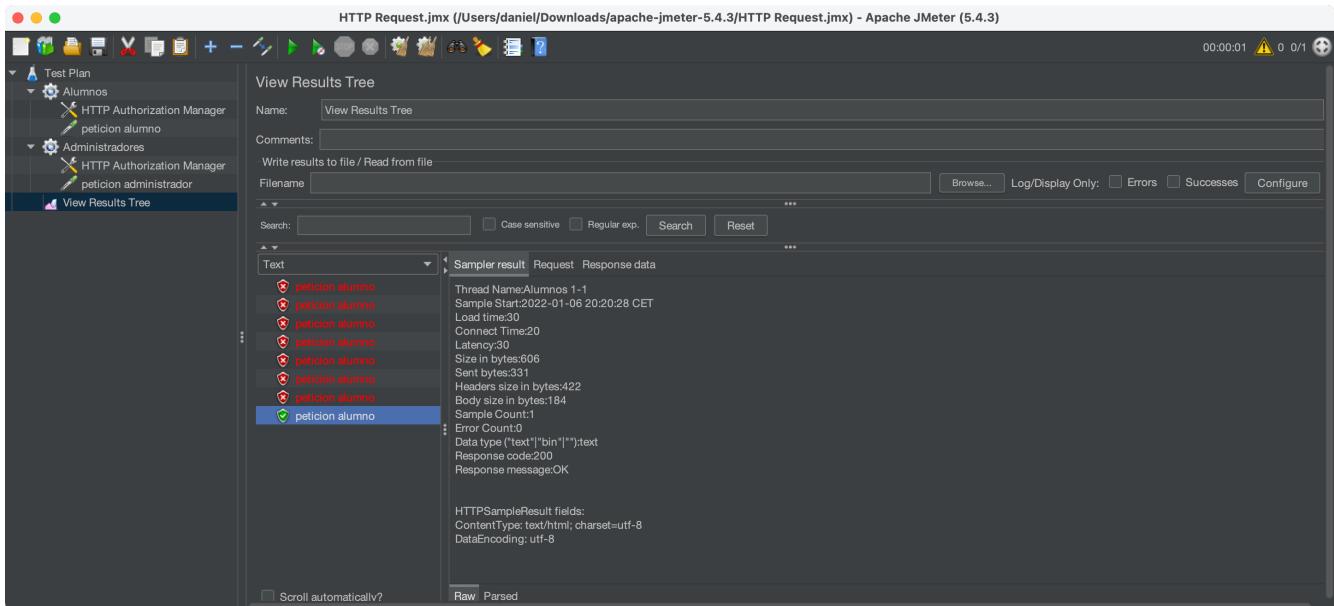


Figure 9: Petición HTTP Correcta

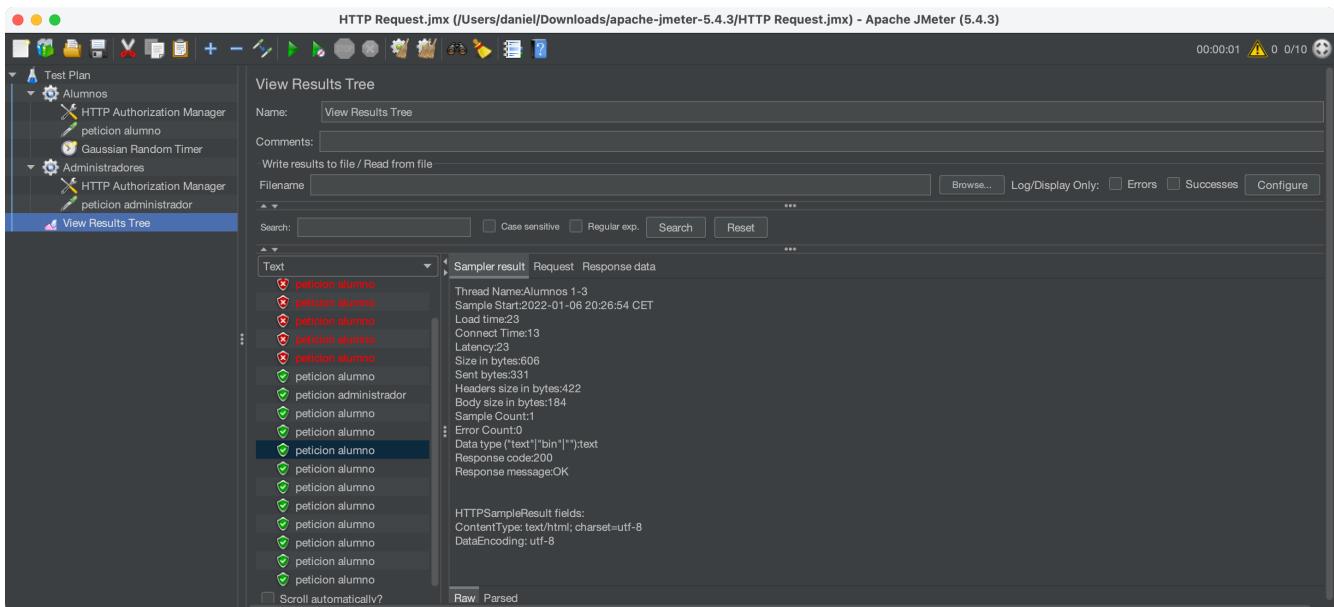


Figure 10: Results Tree para las esperas aleatorias

```

544 2022-01-06 20:47:49,622 INFO o.a.j.t.JMeterThread: Thread is done: Administradores 2-1
545 2022-01-06 20:47:49,622 INFO o.a.j.t.JMeterThread: Thread finished: Administradores 2-1
546 2022-01-06 20:47:50,403 INFO o.a.j.e.J.JSR223 PostProcessor: ****null
547 2022-01-06 20:47:50,614 INFO o.a.j.t.JMeterThread: Thread is done: Alumnos 1-1
548 2022-01-06 20:47:50,614 INFO o.a.j.t.JMeterThread: Thread finished: Alumnos 1-1
549 2022-01-06 20:47:50,616 INFO o.a.j.e.StandardJMeterEngine: Notifying test listeners of end of test
***
```

Figure 11: No recuperamos el token de forma correcta

Un compañero nos dice que en **template** tenemos que tener `$0$`. Buscamos en la documentación y a primera vista no se indica por qué tiene que ser así.

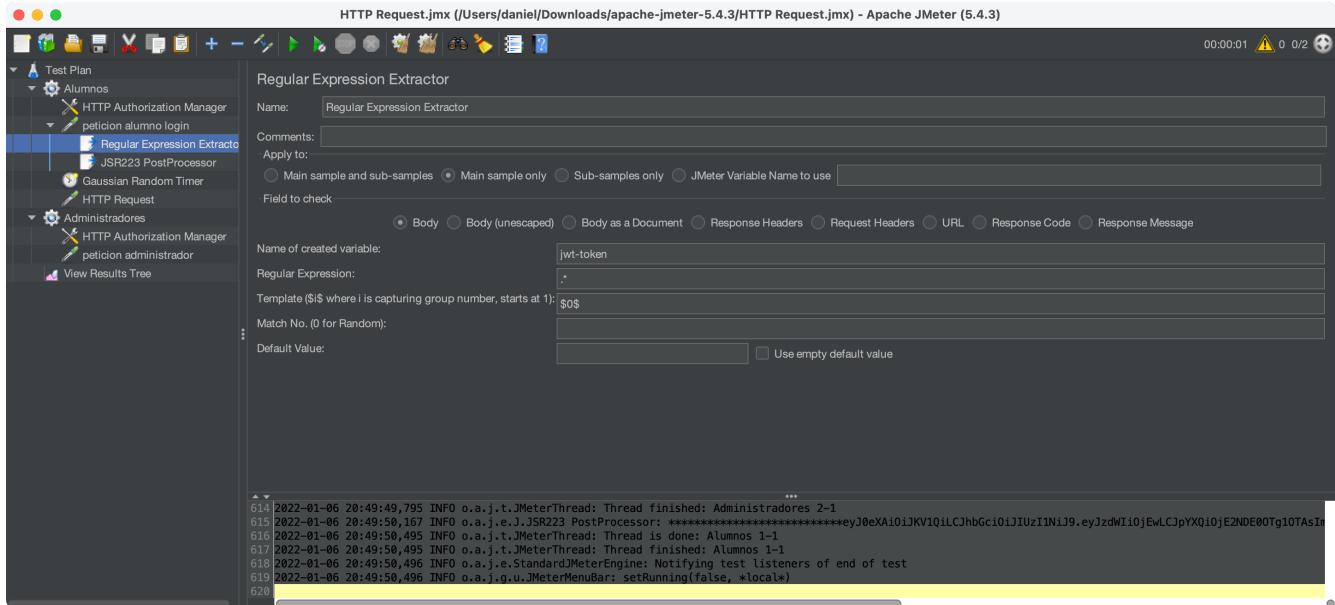


Figure 12: Recuperamos el token de forma correcta

Como se nos dice que hay que añadir el token a la cabecera lo añadimos en un **HTTP Header Manager**. En la página web se nos indica que la autorización tiene que ser mediante el argumento **authorization: Bearer <token>**.

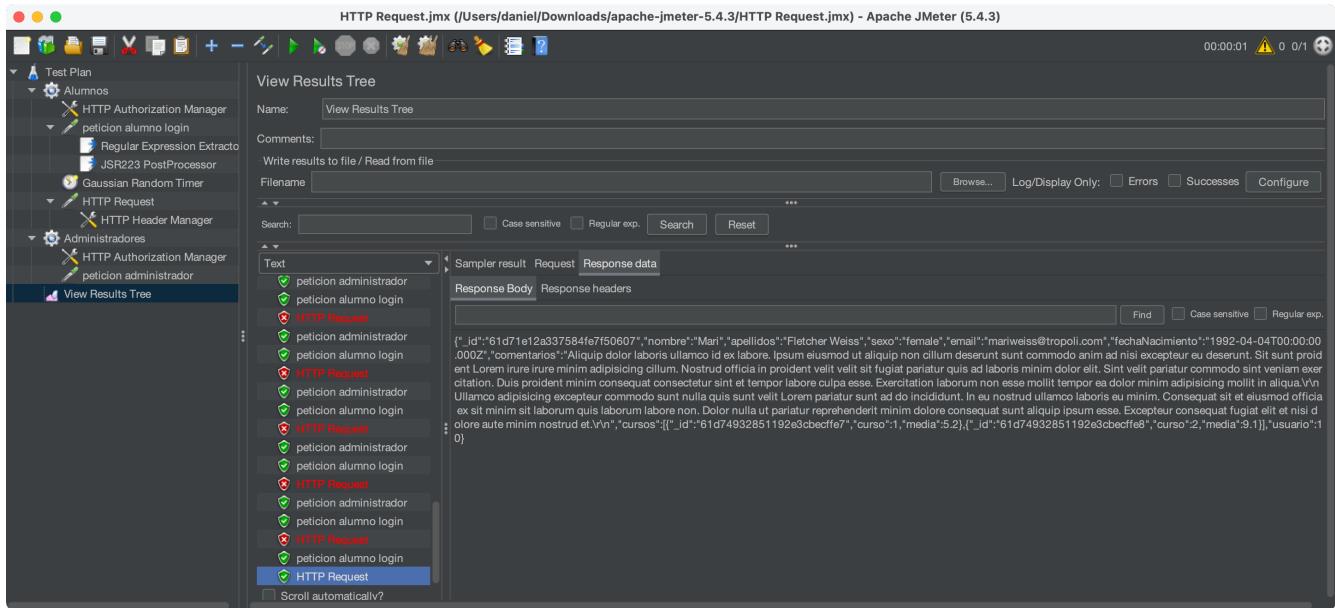


Figure 13: HTTP Request correcto tras añadir el token jwt a la cabecera

## 2.4 (Continuación) El login de alumno, su consulta de datos y login del administrador son peticiones HTTP

Ahora que incluimos el token jwt en las cabeceras tenemos el login del alumno, su consulta de datos y el login del administrador como peticiones HTTP ejecutándose correctamente.

## 2.5 El muestreo para simular el acceso de los administradores lo debe coger el archivo apiAlumnos.log (usando un Acces Log Sampler)

Seguimos los pasos dados en [https://jmeter.apache.org/usermanual/jmeter\\_accesslog\\_sampler\\_step\\_by\\_step.html](https://jmeter.apache.org/usermanual/jmeter_accesslog_sampler_step_by_step.html). Nos encontramos con que, al intentar ejecutar, la API nos indica que no estamos autorizados correctamente. Sospechamos que es porque no hemos incluido el HTTP Header Manager al sampler que hemos añadido.

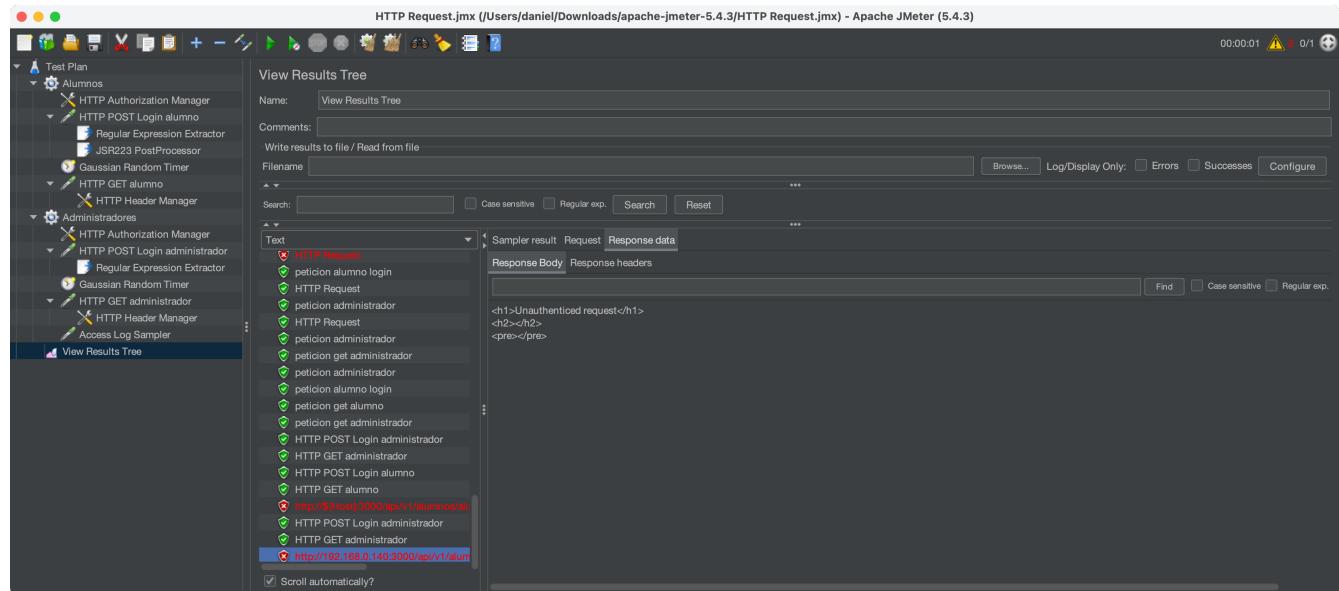


Figure 14: Error al intentar acceder a la API

Ahora sí nos funciona, aunque únicamente nos realiza una petición. Solucionamos esto añadiendo más threads al *Thread group* administradores.

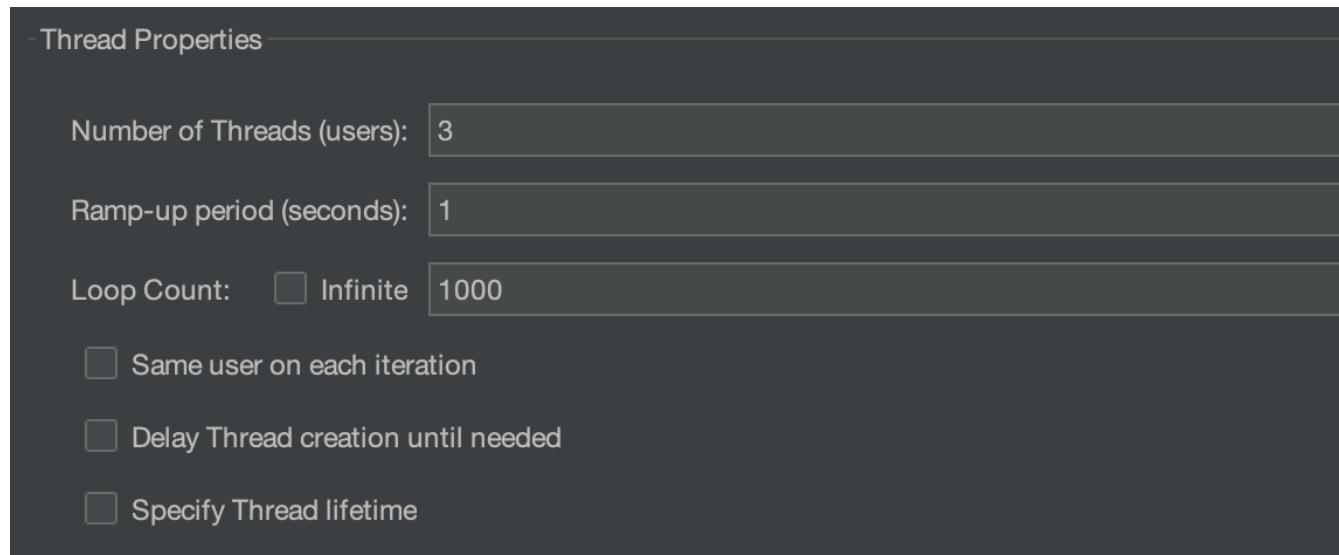


Figure 15: Configuración para el *Thread group* “Administradores”

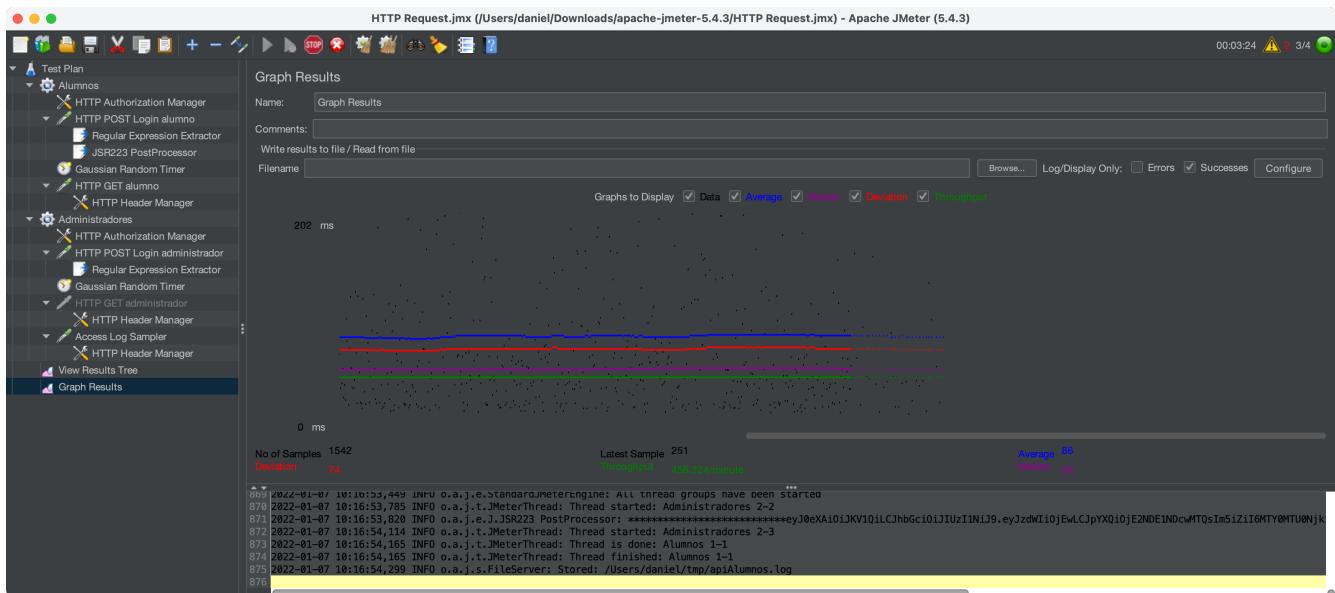


Figure 16: Gráfico para las consultas del Access Log Sampler