

---

# **PRACTICA 4: BENCHMARKING**

## **Y JMETER**

---

Juan Manuel Rodríguez Gómez

Doble Grado en Ingeniería Informática y Matemáticas

Ingeniería de Servidores (Grupo 3)

Curso 2021 – 2022



**UNIVERSIDAD  
DE GRANADA**

## Índice

<b>1. Ejercicio 1.....</b>	1
<b>1.0. Enunciado del Ejercicio.....</b>	1
<b>1.1. Primer Paso: Instalación de Phoronix en CentOS .....</b>	1
<b>1.2. Segundo Paso: Ejecución de los Benchmarks en CentOS.....</b>	5
<b>1.2.1. Ejecución del Benchmark RAMSpeed en CentOS.....</b>	7
<b>1.2.2. Ejecución del Benchmark Sudokut en CentOS .....</b>	12
<b>1.3. Tercer Paso: Instalación de Phoronix en Ubuntu Server .....</b>	15
<b>1.4. Cuarto Paso: Ejecución de los Benchmarks en Ubuntu Server .....</b>	18
<b>1.4.1. Ejecución del Benchmark RAMSpeed en Ubuntu Server .....</b>	18
<b>1.4.2. Ejecución del Benchmark Sudokut en Ubuntu Server .....</b>	25
<b>1.5. Quinto Paso: Comparativa de los Resultados de los Benchmarks.....</b>	27
<b>1.5.1. Comparativa de los Resultados de Ejecución de RAMSpeed.....</b>	27
<b>1.5.2. Comparativa de los Resultados de Ejecución de Sudokut .....</b>	27
<b>2. Ejercicio 2.....</b>	28
<b>2.0. Enunciado del Ejercicio.....</b>	28
<b>2.1. Primer Paso: Instalación de JMeter en nuestra Máquina Anfitrión .....</b>	28
<b>2.2. Segundo Paso: Instalación de Docker en Ubuntu Server .....</b>	30
<b>2.3. Tercer Paso: Instalación de la Aplicación iseP4JMeter en Ubuntu Server.....</b>	36
<b>2.4. Cuarto Paso: Creación del Test en JMeter para la aplicación iseP4JMeter .....</b>	39
<b>2.5. Quinto Paso: Ejecución del Test en JMeter para la aplicación iseP4JMeter.....</b>	52
<b>3. Ejercicio Opcional .....</b>	56
<b>3.0. Enunciado del Ejercicio.....</b>	56
<b>3.1. Primer Paso: Instalación de Docker en nuestra Máquina Anfitrión .....</b>	56
<b>3.2. Segundo Paso: Ejecución del Benchmark Sudokut en el Contenedor Docker con la Imagen de Phoronix.....</b>	61
<b>3.5. Tercer Paso: Comparativa de los Resultados del Benchmark Sudokut .....</b>	65
<b>4. Referencias .....</b>	66

# **1. Ejercicio 1**

## **1.0. Enunciado del Ejercicio**

*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.*

### **1.1. Primer Paso: Instalación de Phoronix en CentOS**

Phoronix es una plataforma que permite la ejecución de un conjunto de benchmarks bajo la agrupación openbenchmarking.org. Esta aplicación puede instalarse a través de los gestores de paquetes correspondientes ya vistos (apt en Ubuntu Server y yum en CentOS). En primer lugar, vamos a instalarlo en CentOS.

Comenzamos conectándonos por SSH a CentOS desde nuestro localhost para así trabajar más cómodos (debido a que para la instalación de Phoronix necesitamos copiar y pegar comandos de gran extensión, y no podemos copiar y pegar desde nuestro localhost a VirtualBox).

```
juanma@Lenovo-JuanmaLinux:~$ ssh juanmaarg6@192.168.56.110 -p 22022
juanmaarg6@192.168.56.110's password:
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Wed May 18 15:09:02 2022
[juanmaarg6@localhost ~]$ █
```

Seguiremos la siguiente web para instalar Phoronix en CentOS:

<https://arstechnica.com/linux/guides/2012/08/how-to-install-the-phoronix-test-suite-on-centos-6-4>

Comenzamos instalando las dependencias necesarias para Phoronix. Para ello, lanzamos el comando `sudo yum install wget php-cli php-xml bzip2`.

```
[juanmaarg6@localhost ~]$ sudo yum install wget php-cli php-xml bzip2
[sudo] password for juanmaarg6:
CentOS-8 - AppStream
CentOS-8 - Extras
Extra Packages for Enterprise Linux Modular 8 - x86_64
Extra Packages for Enterprise Linux Modular 8 - x86_64
Extra Packages for Enterprise Linux 8 - x86_64
Extra Packages for Enterprise Linux 8 - i386
Zabbix Official Repository - x86_64
Zabbix Official Repository non-supported - x86_64
El paquete php-cli-7.2.24-1.module_el8.2.0+313+b04d0ad6.x86_64 ya está instalado.
El paquete bzip2-1.0.6-26.el8.x86_64 ya está instalado.
Dependencias resueltas.

----- Paquete ----- Arquitectura ----- Versión ----- Repositorio ----- Tamaño -----
Instalando:
libxml2           x86_64          7.2.24-1.module_el8.2.0+313+b04d0ad6      AppStream          100 k
wget              x86_64          1.19.5-10.el8                    AppStream          734 k
Instalando dependencias:
libxslt            x86_64          1.1.32-6.el8                     BaseOS            250 k

Resumen de la transacción

Instalar 3 Paquetes

Tamaño total de la descarga: 3.0 M
Tamaño instalado: 3.0 M
¿Está de acuerdo [s/N]?:
```

Obtenemos del repositorio oficial el fichero correspondiente para instalar Phoronix con el siguiente comando:

```
sudo wget https://phoronix-test-suite.com/releases/phoronix-test-suite-8.4.1.tar.gz
```

y descomprimimos el fichero descargado con el comando `sudo tar xvzf phoronix-test-suite-8.4.1.tar.gz`.

```
[juanmaarg6@localhost ~]$ sudo wget https://phoronix-test-suite.com/releases/phoronix-test-suite-8.4.1.tar.gz
2022-05-18 15:12:03 -> [https://phoronix-test-suite.com/releases/phoronix-test-suite-8.4.1.tar.gz]
Resolving https://phoronix-test-suite.com... 192.211.48.82
Connecting to phoronix-test-suite.com (phoronix-test-suite.com)[192.211.48.82]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 873706 (853K) [application/x-gzip]
Grabando a: "phoronix-test-suite-8.4.1.tar.gz"
phoronix-test-suite-8.4.1.tar.gz          100%[=====] 853,23K   915KB/s  en 0,9s
2022-05-18 15:12:05 [873706 / 873706]

[juanmaarg6@localhost ~]$ sudo tar xvzf phoronix-test-suite-8.4.1.tar.gz
phoronix-test-suite/
phoronix-test-suite/deploy/
phoronix-test-suite/deploy/phromantic-upstart/
phoronix-test-suite/deploy/phromantic-upstart/phromantic-server.conf
phoronix-test-suite/deploy/phromantic-upstart/phromantic-client.conf
phoronix-test-suite/deploy/phromantic-upstart/phromantic-client-alt.conf
phoronix-test-suite/deploy/rpm/package/
phoronix-test-suite/deploy/rpm-package/build-package-rpm.php
phoronix-test-suite/deploy/farm-system-customizations/
phoronix-test-suite/deploy/farm-system-customizations/intel-xorg-headless.conf
phoronix-test-suite/deploy/farm-system-customizations/radeon-xorg-headless.conf
phoronix-test-suite/deploy/farm-system-customizations/ubuntu-intlctl-setup.sh
phoronix-test-suite/deploy/deployments.md
phoronix-test-suite/deploy/phromantic-lnttd/
phoronix-test-suite/deploy/phromantic-suid/phromantic-client
```

Luego, nos movemos a la carpeta descomprimida con `cd phoronix-test-suite` y finalmente ejecutamos el script de instalación de Phoronix con `sudo ./install.sh`.

```
[juanmaarg6@localhost ~]$ cd phoronix-test-suite
[juanmaarg6@localhost phoronix-test-suite]$ sudo ./install.sh
which: no xdg-mime in (/sbin:/bin:/usr/sbin:/usr/bin)

Phoronix Test Suite Installation Completed

Executable File: /usr/bin/phoronix-test-suite
Documentation: /usr/share/doc/phoronix-test-suite/
Phoronix Test Suite Files: /usr/share/phoronix-test-suite/

[juanmaarg6@localhost phoronix-test-suite]$
```

Si intentamos ejecutar Phoronix con el comando phoronix-test-suite obtenemos la siguiente salida:

```
[juanmaarg6@localhost phoronix-test-suite]$ phoronix-test-suite
The following PHP extensions are REQUIRED:
JSON      JSON support is required for OpenBenchmarking.org.
```

Luego, tenemos que instalar la extensión JSON de PHP para poder ejecutar Phoronix. Para ello, buscamos el paquete requerido con el comando `yum search json | grep php`.

```
[juanmaarg6@localhost phoronix-test-suite]$ yum search json | grep php
CentOS-8 - AppStream          9.2 MB/s | 8.4 MB   00:00
CentOS-8 - Base                6.5 MB/s | 4.6 MB   00:00
CentOS-8 - Extras              19 kB/s | 10 kB   00:00
Extra Packages for Enterprise Linux Modular 8 - 822 kB/s | 1.0 MB   00:01
Extra Packages for Enterprise Linux 8 - x86_64  6.6 MB/s | 11 MB   00:01
Zabbix Official Repository - x86_64        135 kB/s | 263 kB   00:01
Zabbix Official Repository non-supported - x86_1.2 kB/s | 1.2 kB   00:00
php-json.x86_64 : JavaScript Object Notation extension for PHP
[juanmaarg6@localhost phoronix-test-suite]$
```

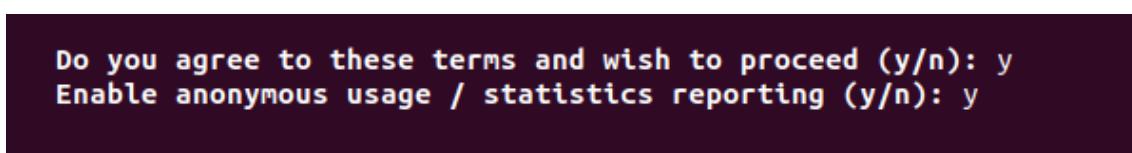
Una vez encontrado el paquete, lo instalamos con `sudo yum install php-json.x86_64`.

```
[juannaarg6@localhost phoronix-test-suite]$ sudo yum install php-json.x86_64
Ultima actualización: 2022-05-16 15:17:01 EDT.
Dependencias resueltas.
=====
Paquete           Arquitectura     Versión           Repositorio      Tamaño
Instalando:
php-json          x86_64           7.2.24-1.module_el8.2.0+313+b04d0a66          AppStream       73 k
Resumen de la transacción
Instalar 1 Paquete
Tamaño total de la descarga: 73 k
Tamaño instalado: 44 k
Tamaño de acuerdo a /dev/urandom: 5
Descargando paquetes:
php-json-7.2.24-1.module_el8.2.0+313+b04d0a66.x86_64.rpm: 221 kB/s | 73 kB   00:00
Total
          Verificando operación de operación
          Verificación de operación exitosa,
          Ejecutando prueba de operaciones
          Prueba de operación exitosa,
          Ejecutando operación
          Instalando
            Instalando : php-json-7.2.24-1.module_el8.2.0+313+b04d0a66.x86_64
            Ejecutando scriptlet: php-json-7.2.24-1.module_el8.2.0+313+b04d0a66.x86_64
            Verificando : php-json-7.2.24-1.module_el8.2.0+313+b04d0a66.x86_64
Instalado:
  php-json-7.2.24-1.module_el8.2.0+313+b04d0a66.x86_64
::listo!
[juannaarg6@localhost phoronix-test-suite]$
```

Si probamos ahora a ejecutar *phoronix-test-suite*, vemos que se ejecuta correctamente.

The screenshot shows a terminal window titled "Phoronix Test Suite v8.4.1 User Agreement". The window contains the "User Agreement & Notices" text, which includes legal disclaimers about the software's source, license, and warranty. It also discusses reporting bugs and feature requests. At the bottom of the text, there is a question: "Do you agree to these terms and wish to proceed (y/n):" followed by a small input field containing a single character.

Tras aceptar los términos, Phoronix nos ofrecerá información de todos los comandos que podemos ejecutar:



This screenshot is identical to the one above, showing the same list of commands and sections for the Phoronix Test Suite. The terminal window has a light background and black text, and it is titled "Phoronix Test Suite v8.4.1 (skiptvet)".

Para ver la información de nuestro sistema usando Phoronix, lanzamos el comando *phoronix-test-suite system-info*.

```
[juanmaarg@localhost phoronix-test-suite]$ phoronix-test-suite system-info

Phoronix Test Suite v8.4.1
AN OUTDATED VERSION OF THE PHORONIX TEST SUITE IS INSTALLED.
THE VERSION IN USE IS 8.4.1 (8410), BUT THE LATEST IS PTS-CORE 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

System Information

PROCESSOR : Intel Core i7-7700HQ
Core Count: 1
Extensions: SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE
Cache Size: 6144 KB

GRAPHICS : VMware SVGA II
Screen: 2048x2048

MOTHERBOARD: Oracle VirtualBox v1.2
BIOS Version: VirtualBox
Chipset: Intel 440FX B2441FX PMC
Audio: Intel 82801AA AC 97 Audio
Network: Intel 82540EM Gigabit

MEMORY : 5120MB

DISK: 2 x 11GB VBOX HD
File-System: xfs
Mount Options: attr2 inode64 noquota relatime rw seclabel
Disk Scheduler: MQ-DEADLINE

OPERATING SYSTEM: CentOS Linux 8
Kernel: 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

[juanmaarg@localhost phoronix-test-suite]$
```

## 1.2. Segundo Paso: Ejecución de los Benchmarks en CentOS

Tenemos una gran variedad de benchmarks para ejecutar en Phoronix. Podemos ver información acerca de todos ellos en la siguiente página web:

<https://openbenchmarking.org/tests>

También podemos ver la lista de benchmarks disponibles con el comando *phoronix-test-suite list-available-tests*.

```
[juanmaarg6@localhost phoronix-test-suite]$ phoronix-test-suite list-available-tests

Phoronix Test Suite v8.4.1
AN OUTDATED VERSION OF THE PHORONIX TEST SUITE IS INSTALLED.
THE VERSION IN USE IS 8.4.1 (8410), BUT THE LATEST IS PTS-CORE 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

Available Tests
pts/ai-benchmark          - AI Benchmark Alpha           System
pts/aircrack-ng            - Aircrack-ng                 Processor
pts/amg                     - Algebraic Multi-Grid Benchmark Processor
pts/aobench                 - AO Bench                  Processor
pts/aom-av1                 - AOM AV1                  Processor
pts/apache                  - Apache HTTP Server        System
pts/apache-siege            - Apache Siege              System
pts/appleseed                - Appleseed                System
pts/arrayfire                - ArrayFire                Processor
pts/askap                    - ASKAP                   System
pts/asmfish                  - asmFish                 Processor
pts/astcenc                  - ASTC Encoder              System
pts/avifenc                  - libavif avifenc          Processor
pts/basemark                 - Basemark GPU             System
pts/basis                    - Basis Universal          System
pts/blake2                  - BLAKE2                  Processor
pts/blender                  - Blender                  System
pts/blogbench                - BlogBench               Disk
pts/blosc                    - C-Blosc                 Processor
pts/bork                     - Bork File Encrypter      Processor
pts/botan                    - Botan                   Processor
pts/brl-cad                  - BRL-CAD                 System
pts/build-apache             - Timed Apache Compilation Processor
pts/build-clash              - Timed Clash Compilation Processor
pts/build-eigen              - Timed Eigen Compilation Processor
pts/build-erlang              - Timed Erlang/OTP Compilation Processor
pts/build-ffmpeg              - Timed FFmpeg Compilation Processor
pts/build-gcc                 - Timed GCC Compilation       Processor
pts/build-gdb                 - Timed GDB GNU Debugger Compilation Processor
pts/build-gem5                - Timed Gem5 Compilation       Processor
pts/build-godot              - Timed Godot Game Engine Compilation Processor
pts/build-imagemagick         - Timed ImageMagick Compilation Processor
pts/build-linux-kernel         - Timed Linux Kernel Compilation Processor
pts/build-llvm                - Timed LLVM Compilation        Processor
pts/build-mesa                - Timed Mesa Compilation        Processor
pts/build-mplayer             - Timed MPlayer Compilation       Processor
pts/build-nodejs              - Timed Node.js Compilation       Processor
pts/build-php                 - Timed PHP Compilation        Processor
pts/build-wasmer              - Timed Wasmer Compilation       Processor
pts/build2                    - Build2                   Processor
pts/bullet                   - Bullet Physics Engine       Processor
pts/byte                     - BYTE Unix Benchmark        Processor
pts/c-ray                     - C-Ray                   Processor
pts/cachebench                - CacheBench              Processor
```

En nuestro caso, hemos elegido ejecutar los siguientes benchmarks en CentOS (y más tarde también en Ubuntu Server) mediante Phoronix:

- RAMSpeed: Benchmark que comprueba el rendimiento de la memoria RAM del sistema.
  - Para ejecutarlo: *phoronix-test-suite benchmark ramspeed*
  - Para más información:  
<https://openbenchmarking.org/test/pts/ramspeed>
  
- Sudokut: Benchmark que comprueba cuánto tarda el sistema en resolver 100 sudokus.
  - Para ejecutarlo: *phoronix-test-suite benchmark sudokut*
  - Para más información:  
<https://openbenchmarking.org/test/pts/sudokut>

## **1.2.1. Ejecución del Benchmark RAMSpeed en CentOS**

Lanzamos el comando *phoronix-test-suite benchmark ramspeed* (así nos bajamos y ejecutamos el test en un solo comando, también podríamos hacerlo en dos comandos, primero nos bajamos el test con *phoronix-test-suite install ramspeed* y luego ejecutarlo con el comando *phoronix-test-suite run ramspeed*. Esto mismo se aplica al resto de benchmarks).

```
[juanmaarg6@localhost phoronix-test-suite]$ phoronix-test-suite benchmark ramspeed

Phoronix Test Suite v8.4.1
AN OUTDATED VERSION OF THE PHORONIX TEST SUITE IS INSTALLED.
THE VERSION IN USE IS 8.4.1 (8410), BUT THE LATEST IS PTS-CORE 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

The following dependencies are needed and will be installed:

- gcc
- gcc-c++
- make
- autoconf
- automake
- glibc-static
- patch
- expat-devel

This process may take several minutes.
[sudo] password for juanmaarg6:
Error: No se pudo encontrar ningún resultado: glibc-static
Última comprobación de caducidad de metadatos hecha hace 0:06:07, el mié 18 may 2022 15:17:01 EDT.
No match for argument: glibc-static

There are dependencies still missing from the system:
- Compiler / Development Libraries

1: Ignore missing dependencies and proceed with installation.
2: Skip installing the tests with missing dependencies.
3: Re-attempt to install the missing dependencies.
4: Quit the current Phoronix Test Suite process.

Missing dependencies action: 4
```

Vemos que nos da error a la hora de instalar el test debido a que faltan dependencias en el sistema, concretamente, nos falta el paquete *glibc-static*. Encontramos la solución a este problema en la siguiente página:

<https://stackoverflow.com/questions/60238534/add-libraries-for-static-compilation-centos-8>

```
[juanmaarg6@localhost phoronix-test-suite]$ sudo dnf config-manager --enable PowerTools
[juanmaarg6@localhost phoronix-test-suite]$ sudo dnf install glibc-static
CentOS-8 - PowerTools           2.9 MB/s | 2.3 MB  00:00
Última comprobación de caducidad de metadatos hecha hace 0:00:01, el mié 18 may 2022 15:24:48 EDT.
Dependencias resueltas.
=====
Paquete      Arq.    Versión       Repositorio   Tam.
=====
Instalando:
glibc-static  x86_64  2.28-164.el8     PowerTools   2.1 M
Actualizando:
glibc          x86_64  2.28-164.el8     BaseOS      3.6 M
glibc-all-langpacks  x86_64  2.28-164.el8     BaseOS      25 M
glibc-common    x86_64  2.28-164.el8     BaseOS      1.3 M
libcrypt        x86_64  4.1.1-6.el8      BaseOS      73 k
Instalando dependencias:
glibc-devel     x86_64  2.28-164.el8     BaseOS      1.0 M
glibc-headers   x86_64  2.28-164.el8     BaseOS      480 k
kernel-headers  x86_64  4.18.0-348.7.1.el8_5  BaseOS      8.3 M
libcrypt-devel   x86_64  4.1.1-6.el8      BaseOS      25 k
libcrypt-static  x86_64  4.1.1-6.el8      PowerTools  56 k
Resumen de la transacción
=====
Instalar  6 Paquetes
Actualizar 4 Paquetes

Tamaño total de la descarga: 42 M
¿Está de acuerdo [s/N]?:
```

Ahora al lanzar el comando `phoronix-test-suite benchmark ramspeed` no obtenemos ningún fallo de dependencias que tengamos que instalar. Esperamos un poco a que el propio programa instale las dependencias necesarias para ejecutar el benchmark.

```
[juanmaarg6@localhost phoronix-test-suite]$ phoronix-test-suite benchmark ramspeed

Phoronix Test Suite v8.4.1
AN OUTDATED VERSION OF THE PHORONIX TEST SUITE IS INSTALLED.
THE VERSION IN USE IS 8.4.1 (8410), BUT THE LATEST IS PTS-CORE 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

The following dependencies are needed and will be installed:

- gcc
- gcc-c++
- make
- autoconf
- automake
- glibc-static
- patch
- expat-devel

This process may take several minutes.
```

Una vez instaladas las dependencias, obtendremos la siguiente salida:

```
To Install: pts/ramspeed-1.4.3
Determining File Requirements .....
Searching Download Caches .....
1 Test To Install
  1 File To Download [0.08MB]
  1MB Of Disk Space Is Needed

pts/ramspeed-1.4.3
Test To Install: 1 of 1
  1 File Needed [0.08 MB]
  Downloading: ramsmp-3.5.0.tar.gz
  Downloading ..... [0.08MB]
  Installation Size: 0.72 MB
  Installing Test @ 15:26:33

RAMspeed SMP 3.5.0:
pts/ramspeed-1.4.3
Memory Test Configuration
  1: Copy
  2: Scale
  3: Add
  4: Triad
  5: Average
  6: Test All Options
  ** Multiple items can be selected, delimit by a comma. **
Type: 1
```

Vamos a ejecutar todos los tests en coma flotante que nos ofrece.

```
RAMspeed SMP 3.5.0:
pts/ramspeed-1.4.3
Memory Test Configuration
  1: Copy
  2: Scale
  3: Add
  4: Triad
  5: Average
  6: Test All Options
  ** Multiple items can be selected, delimit by a comma. **
Type: 6

  1: Integer
  2: Floating Point
  3: Test All Options
  ** Multiple items can be selected, delimit by a comma. **
Benchmark: 2
```

Luego nos preguntará si queremos guardar los resultados en un fichero. En este caso, diremos que sí.

```
Would you like to save these test results (Y/n): y
Enter a name for the result file: benchmark-RAMspeed
Enter a unique name to describe this test run / configuration: benchmark

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: Hola
```

Esperamos a que se ejecute el benchmark y obtenemos los siguientes resultados:

```
RAMspeed SMP 3.5.0:
pts/ramspeed-1.4.3 [Type: Add - Benchmark: Floating Point]
Test 1 of 5
Estimated Trial Run Count:      3
Estimated Test Run-Time:        11 Minutes
Estimated Time To Completion:   55 Minutes [16:22 EDT]
    Started Run 1 @ 15:28:25
    Started Run 2 @ 15:30:00
    Started Run 3 @ 15:31:36

Type: Add - Benchmark: Floating Point:
    19243.16
    19103.24
    19009.65

Average: 19118.68 MB/s
Deviation: 0.61%
```

```
RAMspeed SMP 3.5.0:
pts/ramspeed-1.4.3 [Type: Copy - Benchmark: Floating Point]
Test 2 of 5
Estimated Trial Run Count:      3
Estimated Test Run-Time:        11 Minutes
Estimated Time To Completion:   44 Minutes [16:16 EDT]
    Started Run 1 @ 15:33:21
    Started Run 2 @ 15:35:00
    Started Run 3 @ 15:36:36

Type: Copy - Benchmark: Floating Point:
    16910.27
    17353.51
    16549.69

Average: 16937.82 MB/s
Deviation: 2.38%
```

```
RAMspeed SMP 3.5.0:  
  pts/ramspeed-1.4.3 [Type: Scale - Benchmark: Floating Point]  
    Test 3 of 5  
    Estimated Trial Run Count:      3  
    Estimated Test Run-Time:       11 Minutes  
    Estimated Time To Completion: 33 Minutes [16:10 EDT]  
      Started Run 1 @ 15:38:22  
      Started Run 2 @ 15:39:59  
      Started Run 3 @ 15:41:35  
  
    Type: Scale - Benchmark: Floating Point:  
      16530.64  
      16716.7  
      16808.96  
  
    Average: 16685.43 MB/s  
    Deviation: 0.85%
```

```
RAMspeed SMP 3.5.0:  
  pts/ramspeed-1.4.3 [Type: Triad - Benchmark: Floating Point]  
    Test 4 of 5  
    Estimated Trial Run Count:      3  
    Estimated Test Run-Time:       11 Minutes  
    Estimated Time To Completion: 22 Minutes [16:05 EDT]  
      Started Run 1 @ 15:43:19  
      Started Run 2 @ 15:44:56  
      Started Run 3 @ 15:46:34  
  
    Type: Triad - Benchmark: Floating Point:  
      19086.75  
      18697.65  
      18924.09  
  
    Average: 18902.83 MB/s  
    Deviation: 1.03%
```

```
RAMspeed SMP 3.5.0:  
  pts/ramspeed-1.4.3 [Type: Average - Benchmark: Floating Point]  
    Test 5 of 5  
    Estimated Trial Run Count:      3  
    Estimated Time To Completion: 11 Minutes [15:59 EDT]  
      Started Run 1 @ 15:48:20  
      Started Run 2 @ 15:49:57  
      Started Run 3 @ 15:51:36  
  
    Type: Average - Benchmark: Floating Point:  
      18013.56  
      17504.26  
      17690.98  
  
    Average: 17736.27 MB/s  
    Deviation: 1.45%
```

Tras finalizar su ejecución, nos pedirá si queremos ver los resultados almacenados en el archivo que confirmamos antes que queríamos crear para almacenar los resultados:

```
Do you want to view the text results of the testing (Y/n): Y
benchmark-RAMspeed
Hola

benchmark:

Processor: Intel Core i7-7700HQ (1 Core), Motherboard: Oracle VirtualBox v1.2, Chipset: Intel 440FX 82441FX PMC, Memory: 5120MB, Disk: 2 x 11GB VBOX HDD, Graphics: VMware SVGA II, Audio: Intel 8281AA AC 97 Audio, Network: Intel PRO/100 MT Gigabit
OS: CentOS Linux 8, Kernel: 4.18.0-193.el8.x86_64 (x86_64), File-System: xfs, Screen Resolution: 2048x2048, System Layer: Oracle VMWare

RAMspeed SMP 3.5.0
Type: Add - Benchmark: Floating Point
    MB/s > Higher Is Better
        benchmark .. 19119 |=====

RAMspeed SMP 3.5.0
Type: Copy - Benchmark: Floating Point
    MB/s > Higher Is Better
        benchmark .. 16938 |=====

RAMspeed SMP 3.5.0
Type: Scale - Benchmark: Floating Point
    MB/s > Higher Is Better
        benchmark .. 16685 |=====

RAMspeed SMP 3.5.0
Type: Triad - Benchmark: Floating Point
    MB/s > Higher Is Better
        benchmark .. 18903 |=====

RAMspeed SMP 3.5.0
Type: Average - Benchmark: Floating Point
    MB/s > Higher Is Better
        benchmark .. 17736 |=====
```

Finalmente, nos pedirá si queremos subir nuestros resultados de ejecución a OpenBenchmarking.org. Esto es muy recomendable para ayudar a la comunidad. En este caso le decimos que no, pero en las siguientes ejecuciones de benchmarks sí que lo haremos.

```
Would you like to upload the results to OpenBenchmarking.org (y/n): n
[juanmaarg6@localhost phoronix-test-suite]$
```

En resumen, hemos obtenido los siguientes resultados tras ejecutar el benchmark RAMSpeed en CentOS:

Sistema Operativo	CentOS Linux 8
RAM	5GB – 5120 MB
Benchmark	RAMSpeed
Test 1: Add	19118.68 MB/s
Test 2: Copy	16937.82 MB/s
Test 3: Scale	16685.43 MB/s
Test 4: Triad	18902.83 MB/s
Test 5: Average	17736.27 MB/s

## **1.2.2. Ejecución del Benchmark Sudokut en CentOS**

Lanzamos el comando *phoronix-test-suite benchmark sudokut*.

```
[juanmaarg6@localhost phoronix-test-suite]$ phoronix-test-suite benchmark sudokut
```

Al igual que con el anterior benchmark, esperamos un poco a que el propio programa instale las dependencias necesarias para ejecutarse (esta vez no obtenemos ningún error).

```
Phoronix Test Suite v8.4.1
AN OUTDATED VERSION OF THE PHORONIX TEST SUITE IS INSTALLED.
THE VERSION IN USE IS 8.4.1 (8410), BUT THE LATEST IS PTS-CORE 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

The following dependencies are needed and will be installed:
- tcl

This process may take several minutes.
[sudo] password for juanmaarg6:
Última comprobación de caducidad de metadatos hecha hace 0:35:35, el mié 18 may 2022 15:24:48 EDT.
Dependencias resueltas.
=====
Paquete      Arquitectura    Versión          Repositorio   Tam.
=====
Instalando:
tcl          x86_64           1:8.6.8-2.el8     BaseOS        1.1 M

Resumen de la transacción
=====
Instalar 1 Paquete

Tamaño total de la descarga: 1.1 M
Tamaño instalado: 4.2 M
```

Nos preguntará si queremos guardar los resultados en un fichero. Al igual que antes, diremos que sí.

```
Would you like to save these test results (Y/n): y
Recently Saved Test Results:
benchmark-ramspeed [Today]

Enter a name for the result file: benchmark-sudokut
Enter a unique name to describe this test run / configuration: benchmark2

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: Adios
```

Tras ejecutarse el benchmark obtenemos el siguiente resultado:

```
Sudokut 0.4:  
pts/sudokut-1.0.1  
Test 1 of 1  
Estimated Trial Run Count: 3  
Estimated Time To Completion: 2 Minutes [16:03 EDT]  
Started Run 1 @ 16:01:38  
Started Run 2 @ 16:01:52  
Started Run 3 @ 16:02:06  
  
Total Time:  
13.031840085983  
13.059283971786  
13.116667985916  
  
Average: 13.07 Seconds  
Deviation: 0.33%
```

Nos pide si queremos ver los resultados almacenados en el archivo que confirmamos antes que queríamos crear para almacenar los resultados:

```
Do you want to view the text results of the testing (Y/n): Y  
benchmark-sudokut  
Adlos  
  
benchmark2:  
  
Processor: Intel Core i7-7700HQ (1 Core), Motherboard: Oracle VirtualBox v1.2, Chipset: Intel 440FX 82441FX PMC, Memory: 5120MB, Dtsk: 2 x 11GB VBOX HDD, Graphics: VMware SVGA II, Audio: Intel 82801AA AC 97 Audio, Network: Intel 82540EM Gigabit  
OS: CentOS Linux 8, Kernel: 4.18.0-193.el8.x86_64 (x86_64), File-System: xfs, Screen Resolution: 2048x2048, System Layer: Oracle VMWare  
Sudokut: 0.4  
Total Time  
Seconds < Lower Is Better  
benchmark2 ... 13.07 |=====
```

Finalmente, nos pide si queremos subir nuestros resultados de ejecución a OpenBenchmarking.org. Esta vez, decimos que sí, y nos devuelve el enlace con los resultados de nuestra ejecución:

<https://openbenchmarking.org/result/2205186-SK-BENCHMARK50>

```
Would you like to upload the results to OpenBenchmarking.org (y/n): y  
Would you like to attach the system logs (lspci, dmesg, lsusb, etc) to the test result (y/n): y  
  
Results Uploaded To: https://openbenchmarking.org/result/2205186-SK-BENCHMARK50  
[juanmaarg6@localhost phoronix-test-suite]$
```

Benchmark-sudokut Per... X https://openbenchmarking.org/result/2205186-SK-BENCHMARK50

OpenBenchmarking.org Tests Suites Latest Results

benchmark-sudokut

Search Register Login Export Benchmark Data

Popular Tests  
Timed Linux Kernel Compilation  
Blender  
SVT-AV1  
7-Zip Compression  
Huffman  
Flexible IO Tester  
Newest Tests  
fast-cli  
speedtest-cli  
OSPray Studio  
SPECCviewPerf 2020  
F1 2021  
Timed Gem5 Compilation  
Recently Updated Tests  
Stress-NG  
Blender  
WebP2 Image Encode  
SVT-AV1  
Ethr  
Apache HTTP Server  
New & Recently Updated Tests  
Recently Updated Suites  
Internet Speed  
CPU / Processor Suite  
Raytracing  
New & Recently Updated Suites  
Currently Trending Results  
f107052022  
AMD EPYC 7302 / 7402 / 7502 / 7742 2P vs. Xeon Benchmarks

Statistics Graph Settings Table  
 Remove Outliers Before Calculating Averages  Prefer Vertical Bar Graphs  Show Detailed System Result Table  
 Run Management  
 RESULT VIEW LOGS PERFORMANCE PER DOLLAR DATE TEST  
 IDENTIFIER benchmark2 View System Logs RUN DURATION  
 May 18 1 Minute  
 Refresh Results

### benchmark-sudokut

**OpenBenchmarking.org Phoronix Test Suite 10.8.3**

<b>Intel Core i7-7700HQ (1 Core)</b>	Processor
<b>Oracle VirtualBox v1.2</b>	Motherboard
<b>Intel 440FX 82441FX PMC</b>	Chipset
<b>5120MB</b>	Memory
<b>2 x 11GB VBOX HDD</b>	Disk
<b>VMware SVGA II</b>	Graphics
<b>Intel 82801AA AC 97 Audio</b>	Audio
<b>Intel 82540EM Gigabit</b>	Network

Privacy

Benchmark-sudokut Per... X https://openbenchmarking.org/result/2205186-SK-BENCHMARK50

recently updated suites  
Internet Speed  
CPU / Processor Suite  
Raytracing  
New & Recently Updated Suites  
Currently Trending Results  
f107052022  
AMD EPYC 7302 / 7402 / 7502 / 7742 2P vs. Xeon Benchmarks  
AMD Ryzen 5 2600 / Ryzen 7 2700 Ubuntu Linux Benchmarks  
Radeon RX 6750 XT Linux Review Benchmarks  
firmv2  
20220517-ktliper1  
Component Benchmarks  
CPUs / Processors  
GPUs / Graphics  
OpenGL  
Disks / Storage  
Motherboards  
File-Systems  
Operating Systems  
OpenBenchmarking.org  
Corporate / Organization Info  
Bug Reports / Feature Requests

**Intel 440FX 82441FX PMC** Chipset  
**5120MB** Memory  
**2 x 11GB VBOX HDD** Disk  
**VMware SVGA II** Graphics  
**Intel 82801AA AC 97 Audio** Audio  
**Intel 82540EM Gigabit** Network  
**CentOS Linux 8** OS  
**4.18.0-193.el8.x86\_64 (x86\_64)** Kernel  
**xfs** File-System  
**2048x2048** Screen Resolution  
**Oracle VMware** System Layer

**Benchmark-sudokut Performance**  
 - SELinux + KPTI + usercopy/swaps barriers and \_\_user pointer sanitization + Full generic retpoline STIBP: disabled RSB filling + PTE Inversion

System Logs

Sudoku

This is a test of Sudokut, which is a Sudoku puzzle solver written in Tcl. This test measures how long it takes to solve 100 Sudoku puzzles. Learn more via the OpenBenchmarking.org test page.

**Sudokut 0.4**  
 Total Time  
 Seconds, Lower is Better  
 13.07  
 10 ± 0.02, N = 1

ptsli OpenBenchmarking.org

Privacy

En resumen, hemos obtenido los siguientes resultados tras ejecutar el benchmark Sudokut en CentOS:

<b>Sistema Operativo</b>	<i>CentOS Linux 8</i>
<b>Benchmark</b>	<i>Sudokut</i>
<b>Tiempo medio resultante</b>	<b>13.07 segundos</b>
<b>Desviación típica</b>	<b>0.33%</b>

### **1.3. Tercer Paso: Instalación de Phoronix en Ubuntu Server**

Procedemos igual que en CentOS. Comenzamos conectándonos por SSH a Ubuntu Server desde nuestro localhost.

```
juanma@Lenovo-JuanmaLinux:~$ ssh juanmaarg6@192.168.56.105 -p 22022
juanmaarg6@192.168.56.105's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-109-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information disabled due to load higher than 1.0

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation

9 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed May 18 20:10:23 2022
juanmaarg6@ise-ubuntu:~$ █
```

Seguiremos la siguiente web para instalar Phoronix en Ubuntu Server:

<https://ubunlog.com/phoronix-test-suite-una-herramienta-para-benchmark-multiplataforma/>

Obtenemos del repositorio oficial el fichero correspondiente para instalar Phoronix con el siguiente comando:

```
wget http://phoronix-test-suite.com/releases/repo/pts.debian/files/phoronix-test-suite_8.6.0_
all.deb
```

y realizamos la instalación con *sudo dpkg -i phoronix-test-suite\_8.6.0\_all.deb*.

```

juanmaarg6@ise-ubuntu:~$ wget http://phoronix-test-suite.com/releases/repo/pts.debian/files/phoronix-test-suite_8.6.0_all.deb
--2022-05-18 20:11:30-- http://phoronix-test-suite.com/releases/repo/pts.debian/files/phoronix-test-suite_8.6.0_all.deb
Resolving phoronix-test-suite.com (phoronix-test-suite.com)... 192.211.48.82
Connecting to phoronix-test-suite.com (phoronix-test-suite.com)|192.211.48.82|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 603468 (589K) [application/x-debian-package]
Saving to: 'phoronix-test-suite_8.6.0_all.deb'

phoronix-test-suite_8.6.0_all.deb          100%[=====] 589.32K   573KB/s   in 1.0s

2022-05-18 20:11:38 (573 KB/s) - 'phoronix-test-suite_8.6.0_all.deb' saved [603468/603468]

juanmaarg6@ise-ubuntu:~$ sudo dpkg -i phoronix-test-suite_8.6.0_all.deb
[sudo] password for juanmaarg6:
Selecting previously unselected package phoronix-test-suite.
(Reading database ... 84981 files and directories currently installed.)
Preparing to unpack phoronix-test-suite_8.6.0_all.deb ...
Unpacking phoronix-test-suite (8.6.0) ...
Setting up phoronix-test-suite (8.6.0) ...
Processing triggers for mime-support (3.64ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for man-db (2.9.1-1) ...
juanmaarg6@ise-ubuntu:~$ 

```

Finalmente, solucionamos cualquier problema con las dependencias con el comando *sudo apt -f install*:

```

juanmaarg6@ise-ubuntu:~$ sudo apt -f install
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 16 not upgraded.
juanmaarg6@ise-ubuntu:~$ 

```

Probamos ahora a ejecutar *phoronix-test-suite* y vemos que se ejecuta correctamente.

```

juanmaarg6@ise-ubuntu:~$ phoronix-test-suite list-available-tests
NOTICE: The following PHP extensions are OPTIONAL but recommended:
BZip2      The bzcompress/bzip2 support can be used for greater file compression.
SQLite3    SQLite3 is required when running a Phoromatic server.
CURL       CURL is recommended for an enhanced download experience.

Phoronix Test Suite V8.6.0
User Agreement

Phoronix Test Suite User Agreement / Notices:

- The Phoronix Test Suite is open-source and licensed under the GNU GPLv3. A copy of the GPLv3 license is included with this software or can also be found at https://www.gnu.org/licenses/gpl-3.0.en.html. However, some tests supported by the Phoronix Test Suite are not open-source software or require commercial software packages.

- The Phoronix Test Suite contains tests which may stress your system and in some cases could exhibit stability problems of the system's hardware or software configuration. The Phoronix Test Suite is provided WITHOUT ANY WARRANTY. In no event shall OpenBenchmarking.org, Phoromatic, Phoronix Media, the Phoronix Test Suite, or any associated stakeholder be liable to any party for any direct or indirect damages for any use of OpenBenchmarking.org -- including, without limitation, any lost profits, business interruption, loss of programs, loss of programmed data, or otherwise.

- If you opt to submit your test results to OpenBenchmarking.org, the final results as well as basic hardware and software details (what is shown in the results viewer) will be shared and publicly accessible through https://www.openbenchmarking.org/ and optionally relevant system details like dmesg and /proc/cpuinfo.

- Anonymous usage reporting / statistics: If enabling the anonymous usage reporting / statistics feature, some information about the Phoronix Test Suite runs will be submitted to OpenBenchmarking.org. This information is used for analytical purposes, including the determining of the most popular tests / suites and calculating average run-times for different test profiles. The test results are not reported in this process nor the installed software / hardware information, but ambient information about the testing process. This information is stored anonymously. More information on this feature is available with the included documentation.

- Community support / public bug reports, feature requests, and other issues can be brought up via GitHub: https://github.com/phoronix-test-suite For enterprise/commercial support, sponsorship, or other professional inquiries, contact commercial@phoronix-test-suite.com.

For more information on the Phoronix Test Suite and its features, visit https://www.phoronix-test-suite.com/, https://github.com/phoronix-test-suite/phoronix-test-suite or view the included documentation or by contacting support@phoronix-test-suite.com. Commercial support, custom engineering, and other services are available by contacting us.

Do you agree to these terms and wish to proceed (y/n): 

```

Tras aceptar los términos, Phoronix nos ofrecerá información de todos los comandos que podemos ejecutar:

```
Do you agree to these terms and wish to proceed (y/n): y
Enable anonymous usage / statistics reporting (y/n): y
```

Probamos a ver la lista de benchmarks disponibles con el comando *phoronix-test-suite list-available-tests*.

```
juanmaarg6@ise-ubuntu:~$ phoronix-test-suite list-available-tests

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 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

Available Tests

pts/ai-benchmark          - AI Benchmark Alpha           System
pts/aircrack-ng            - Aircrack-ng                 Processor
pts/amg                     - Algebraic Multi-Grid Benchmark Processor
pts/aobench                 - AO Bench                  Processor
pts/aom-av1                 - AOM AV1                  Processor
pts/apache                  - Apache HTTP Server        System
pts/apache-siege            - Apache Siege              System
pts/appleseed                - Appleseed                System
pts/arrayfire                - ArrayFire                Processor
pts/askap                    - ASKAP                   System
pts/asmfish                  - asmFish                 Processor
pts/astcenc                  - ASTC Encoder              System
pts/avifenc                  - libavif avifenc          Processor
pts/basemark                 - Basemark GPU             System
pts/basis                    - Basis Universal          System
pts/blake2                  - BLAKE2                  Processor
pts/blender                  - Blender                  System
pts/blogbench                - BlogBench               Disk
pts/blosc                    - C-Blosc                 Processor
pts/bork                     - Bork File Encrypter      Processor
pts/botan                    - Botan                   Processor
pts/brl-cad                  - BRL-CAD                 System
pts/build-apache             - Timed Apache Compilation Processor
pts/build-clash               - Timed Clash Compilation Processor
pts/build-eigen               - Timed Eigen Compilation Processor
pts/build-erlang              - Timed Erlang/OTP Compilation Processor
pts/build-ffmpeg              - Timed FFmpeg Compilation Processor
pts/build-gcc                 - Timed GCC Compilation       Processor
pts/build-gdb                 - Timed GDB GNU Debugger Compilation Processor
pts/build-gems                - Timed Gem5 Compilation       Processor
pts/build-godot               - Timed Godot Game Engine Compilation Processor
pts/build-imagemagick         - Timed ImageMagick Compilation Processor
pts/build-linux-kernel         - Timed Linux Kernel Compilation Processor
pts/build-llvm                 - Timed LLVM Compilation       Processor
pts/build-mesa                 - Timed Mesa Compilation       Processor
pts/build-mplayer              - Timed MPlayer Compilation       Processor
pts/build-nodejs              - Timed Node.js Compilation       Processor
pts/build-php                 - Timed PHP Compilation       Processor
pts/build-wasmer              - Timed Wasmer Compilation       Processor
pts/build2                    - Build2                   Processor
pts/bullet                    - Bullet Physics Engine       Processor
pts/byte                      - BYTE Unix Benchmark        Processor
pts/c-ray                     - C-Ray                   Processor
pts/cachebench                - CacheBench              Processor
pts/caffe                     - Caffe                  System
```

## **1.4. Cuarto Paso: Ejecución de los Benchmarks en Ubuntu Server**

Vamos a ejecutar mediante Phoronix los mismos benchmarks que ejecutamos en CentOS para así poder realizar una comparativa posteriormente:

- RAMSpeed: Benchmark que comprueba el rendimiento de la memoria RAM del sistema.
  - Para ejecutarlo: *phoronix-test-suite benchmark ramspeed*
  - Para más información:  
<https://openbenchmarking.org/test/pts/ramspeed>
- Sudokut: Benchmark que comprueba cuánto tarda el sistema en resolver 100 sudokus.
  - Para ejecutarlo: *phoronix-test-suite benchmark sudokut*
  - Para más información:  
<https://openbenchmarking.org/test/pts/sudokut>

### **1.4.1. Ejecución del Benchmark RAMSpeed en Ubuntu Server**

Lanzamos el comando *phoronix-test-suite benchmark ramspeed* y obtenemos el siguiente error:

```
juannaarg@lse-ubuntu:~$ phoronix-test-suite benchmark ramspeed

[PROBLEM] Failed to find ZIP support for extracting file: /home/juannaarg/.phoronix-test-suite/openbenchmarking.org/pts/ramspeed-1.4.3.zip; install PHP ZIP support or the unzip utility.
Updated OpenBenchmarking.org Repository Index
pts: 466 Distinct Tests, 95 Suites

[PROBLEM] Failed to find ZIP support for extracting file: /home/juannaarg/.phoronix-test-suite/openbenchmarking.org/pts/ramspeed-1.4.3.zip; install PHP ZIP support or the unzip utility.
[PROBLEM] Invalid Argument: ramspeed

CORRECT SYNTAX:
phoronix-test-suite benchmark [Test / Suite / OpenBenchmarking ID / Test Result] ...

Popular Tests:
apache          primeSteve      build-mplayer    build-llvm      npb          stream        build-linux-kernel  build-gcc      compress-7zip
compress-pbzip2  pgbench         tachyon        flo           clomp        gcrypt        rodinia       blake2        fs-mark

Recent OpenBenchmarking.org Results From This IP:
2A519E-SK-BENCHMARKS0 benchmark-sudokut

Possible Suggestions:
- ramspeed [Test]

See available tests to run by visiting OpenBenchmarking.org or running:
phoronix-test-suite list-tests

Tests can be installed by running:
phoronix-test-suite install <test-name>

juannaarg@lse-ubuntu:~$
```

Encontramos la solución a dicho error en el siguiente enlace (básicamente, tenemos que instalar el paquete *unzip*):

```
juanmaarg6@ise-ubuntu:~$ sudo apt install unzip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 16 not upgraded.
Need to get 169 kB of archives.
After this operation, 593 kB of additional disk space will be used.
Get:1 http://es.archive.ubuntu.com/ubuntu focal/main amd64 unzip amd64 6.0-25ubuntu1 [169 kB]
Fetched 169 kB in 1s (172 kB/s)
Selecting previously unselected package unzip.
(Reading database ... 85565 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-25ubuntu1_amd64.deb ...
Unpacking unzip (6.0-25ubuntu1) ...
Setting up unzip (6.0-25ubuntu1) ...
Processing triggers for mime-support (3.64ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
juanmaarg6@ise-ubuntu:~$
```

Ahora al lanzar el comando *phoronix-test-suite benchmark ramspeed* no obtenemos ningún fallo. Esperamos un poco a que el propio programa instale las dependencias necesarias para ejecutar el benchmark.

```
juanmaarg6@ise-ubuntu:~$ phoronix-test-suite benchmark ramspeed

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 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.
```

The following dependencies are needed and will be installed:

- build-essential
- autoconf
- mesa-utils
- unzip
- apt-file

This process may take several minutes.

Una vez instaladas las dependencias, obtendremos la siguiente salida:

```
pts/ranspeed-1.4.3:  
  Test Installation 1 of 1  
  1 File Needed [0.08 MB]  
  Downloading: ransmp-3.5.0.tar.gz  
  Downloading .....  
  Installation Size: 0.72 MB  
  Installing Test @ 20:39:05  
  
RAMspeed SMP 3.5.0:  
pts/ranspeed-1.4.3  
Memory Test Configuration  
1: Copy  
2: Scale  
3: Add  
4: Triad  
5: Average  
6: Test All Options  
** Multiple items can be selected, delimit by a comma. **  
Type: 1
```

Al igual que en CentOS, vamos a ejecutar todos los tests en coma flotante que nos ofrece.

```
RAMspeed SMP 3.5.0:  
pts/ranspeed-1.4.3  
Memory Test Configuration  
1: Copy  
2: Scale  
3: Add  
4: Triad  
5: Average  
6: Test All Options  
** Multiple items can be selected, delimit by a comma. **  
Type: 6  
  
1: Integer  
2: Floating Point  
3: Test All Options  
** Multiple items can be selected, delimit by a comma. **  
Benchmark: 2
```

Luego nos preguntará si queremos guardar los resultados en un fichero. Decimos que sí.

```
System Information  
  
PROCESSOR: Intel Core i7-7700HQ  
Core Count: 1  
Extensions: SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE  
Cache Size: 6144 KB  
  
GRAPHICS: VMware SVGA II  
Screen: 2048x2048  
  
MOTHERBOARD: Oracle VirtualBox v1.2  
BIOS Version: VirtualBox  
Chipset: Intel 440FX 82441FX PMC  
Audio: Intel 82801AA AC 97 Audio  
Network: 2 x Intel 82540EM  
  
MEMORY: 5120MB  
  
DISK:  
File-System: ext4  
Mount Options: relatime rw  
Disk Scheduler: MQ-DEADLINE  
  
OPERATING SYSTEM: Ubuntu 20.04  
Kernel: 5.4.0-109-generic (x86_64)  
Compiler: GCC 9.4.0  
System Layer: Oracle VMware  
Security: KPTI  
+ usercopy/swaps barriers and __user pointer sanitization  
+ Retpolines STIBP: disabled RSB filling  
+ PTE Inversion  
  
Would you like to save these test results (Y/n): y  
Enter a name for the result file: benchmark-ranspeed-ubuntu  
Enter a unique name to describe this test run / configuration: benchmark3  
  
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.  
New Description:
```

Esperamos a que se ejecute el benchmark y obtenemos los siguientes resultados:

```
RAMspeed SMP 3.5.0:  
pts/ramspeed-1.4.3 [Type: Add - Benchmark: Floating Point]  
Test 1 of 5  
Estimated Trial Run Count: 3  
Estimated Test Run-Time: 6 Minutes  
Estimated Time To Completion: 26 Minutes [21:56 UTC]  
Started Run 1 @ 21:31:05  
Started Run 2 @ 21:32:49  
Started Run 3 @ 21:34:30  
  
Type: Add - Benchmark: Floating Point:  
18137.36  
18478.72  
18949.23  
  
Average: 18521.77 MB/s  
Deviation: 2.20%
```

```
RAMspeed SMP 3.5.0:  
pts/ramspeed-1.4.3 [Type: Copy - Benchmark: Floating Point]  
Test 2 of 5  
Estimated Trial Run Count: 3  
Estimated Test Run-Time: 6 Minutes  
Estimated Time To Completion: 21 Minutes [21:56 UTC]  
Started Run 1 @ 21:36:21  
Started Run 2 @ 21:38:03  
Started Run 3 @ 21:39:44  
  
Type: Copy - Benchmark: Floating Point:  
16351.86  
16375.01  
16054.81  
  
Average: 16260.56 MB/s  
Deviation: 1.10%
```

```
RAMspeed SMP 3.5.0:  
pts/ramspeed-1.4.3 [Type: Scale - Benchmark: Floating Point]  
Test 3 of 5  
Estimated Trial Run Count: 3  
Estimated Test Run-Time: 6 Minutes  
Estimated Time To Completion: 16 Minutes [21:56 UTC]  
Started Run 1 @ 21:41:31  
Started Run 2 @ 21:43:10  
Started Run 3 @ 21:44:51  
  
Type: Scale - Benchmark: Floating Point:  
16002.88  
15416.22  
15893.53  
  
Average: 15770.88 MB/s  
Deviation: 1.98%
```

```

RAMspeed SMP 3.5.0:
  pts/ramspeed-1.4.3 [Type: Triad - Benchmark: Floating Point]
Test 4 of 5
Estimated Trial Run Count:    3
Estimated Test Run-Time:      6 Minutes
Estimated Time To Completion: 11 Minutes [21:56 UTC]
    Started Run 1 @ 21:46:38
    Started Run 2 @ 21:48:16
    Started Run 3 @ 21:49:54

Type: Triad - Benchmark: Floating Point:
  19263.47
  19186.54
  18471.25

Average: 18973.75 MB/s
Deviation: 2.30%

```

```

RAMspeed SMP 3.5.0:
  pts/ramspeed-1.4.3 [Type: Average - Benchmark: Floating Point]
Test 5 of 5
Estimated Trial Run Count:    3
Estimated Time To Completion: 6 Minutes [21:56 UTC]
    Started Run 1 @ 21:51:44
    Started Run 2 @ 21:53:24
    Started Run 3 @ 21:55:05

Type: Average - Benchmark: Floating Point:
  17546.91
  17441.81
  17222.21

Average: 17403.64 MB/s
Deviation: 0.95%

```

Tras finalizar su ejecución, nos pedirá si queremos ver los resultados almacenados en el archivo que confirmamos antes que queríamos crear para almacenar los resultados:

```

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

benchmark3:
Processor: Intel Core i7-7700HQ (3 Core), Motherboard: Oracle VirtualBox v1.2, Chipset: Intel 440FX 82441FX PMC, Memory: 5120MB, Disk: 2 x 11GB VBOX HDD, Graphics: VMware SVGA II, Audio: Intel 82801AA AC 97 Audio, Network: 2 x Intel PRO/100 MT Desktop Adapter
OS: Ubuntu 20.04, Kernel: 5.4.0-189-generic (x86_64), Compiler: GCC 9.4.0, File-System: ext4, Screen Resolution: 2048x2048, System Layer: Oracle VMWare

RAMspeed SMP 3.5.0
Type: Add - Benchmark: Floating Point
MB/s > Higher Is Better
benchmark3 . 18522 |=====

RAMspeed SMP 3.5.0
Type: Copy - Benchmark: Floating Point
MB/s > Higher Is Better
benchmark3 . 16261 |=====

RAMspeed SMP 3.5.0
Type: Scale - Benchmark: Floating Point
MB/s > Higher Is Better
benchmark3 . 15771 |=====

RAMspeed SMP 3.5.0
Type: Triad - Benchmark: Floating Point
MB/s > Higher Is Better
benchmark3 . 18074 |=====

RAMspeed SMP 3.5.0
Type: Average - Benchmark: Floating Point
MB/s > Higher Is Better
benchmark3 . 17404 |=====
```

Finalmente, nos pide si queremos subir nuestros resultados de ejecución a OpenBenchmarking.org. Decimos que sí, y nos devuelve el enlace con los resultados de nuestra ejecución:

<https://openbenchmarking.org/result/2205186-SP-BENCHMARK71>

```
Would you like to upload the results to OpenBenchmarking.org (y/n): y
Would you like to attach the system logs (lspci, dmesg, lsusb, etc) to the test result (y/n): y
Results Uploaded To: https://openbenchmarking.org/result/2205186-SP-BENCHMARK71
```

The screenshot shows the OpenBenchmarking.org results page for the benchmark-ramspeed-ubuntu. The page header includes links for Tests, Suites, and Latest Results, along with Search, Register, and Login buttons. The main content area displays the following information:

- Test Suite:** Oracle VMWare testing on Ubuntu 20.04 via the Phoronix Test Suite.
- Comparison:** Compare your own system(s) to this result file with the Phoronix Test Suite by running the command: `phoronix-test-suite benchmark 2205186-SP-BENCHMARK71`.
- Run Management:** Options to Remove Outliers Before Calculating Averages, Prefer Vertical Bar Graphs, Condense Multi-Option Tests Into Single Result Graphs, and Show Detailed System Result Table.
- Statistics:** Shows the date of the run as May 18 25 Minutes.
- Hardware Components:**

Processor	Intel Core i7-7700HQ (1 Core)
Motherboard	Oracle VirtualBox v1.2
Chipset	Intel 440FX 82441FX PMC
Memory	5120MB
Disk	2 x 11GB VBOX HDD
Graphics	VMware SVGA II
Audio	Intel 82801AA AC 97 Audio
Network	2 x Intel 82540EM
OS	Ubuntu 20.04
Kernel	5.4.0-109-generic (x86_64)
- Benchmarks:** Shows the configuration for the benchmark: Intel Core i7-7700HQ (1 Core), Oracle VirtualBox v1.2, Intel 440FX 82441FX PMC, 5120MB, 2 x 11GB VBOX HDD, VMware SVGA II, Intel 82801AA AC 97 Audio, 2 x Intel 82540EM, Ubuntu 20.04, and 5.4.0-109-generic (x86\_64).

The screenshot shows the OpenBenchmarking.org results page for the benchmark-ramspeed-ubuntu, focusing on the performance parameters and command-line options. The page header and sidebar are identical to the previous screenshot.

**Performance Parameters:**

Processor	Intel Core i7-7700HQ (1 Core)
Motherboard	Oracle VirtualBox v1.2
Chipset	Intel 440FX 82441FX PMC
Memory	5120MB
Disk	2 x 11GB VBOX HDD
Graphics	VMware SVGA II
Audio	Intel 82801AA AC 97 Audio
Network	2 x Intel 82540EM
OS	Ubuntu 20.04
Kernel	5.4.0-109-generic (x86_64)
Compiler	GCC 9.4.0
File-System	ext4
Screen Resolution	2048x2048
System Layer	Oracle VMware

**Benchmark-ramspeed-ubuntu Performance Options:**

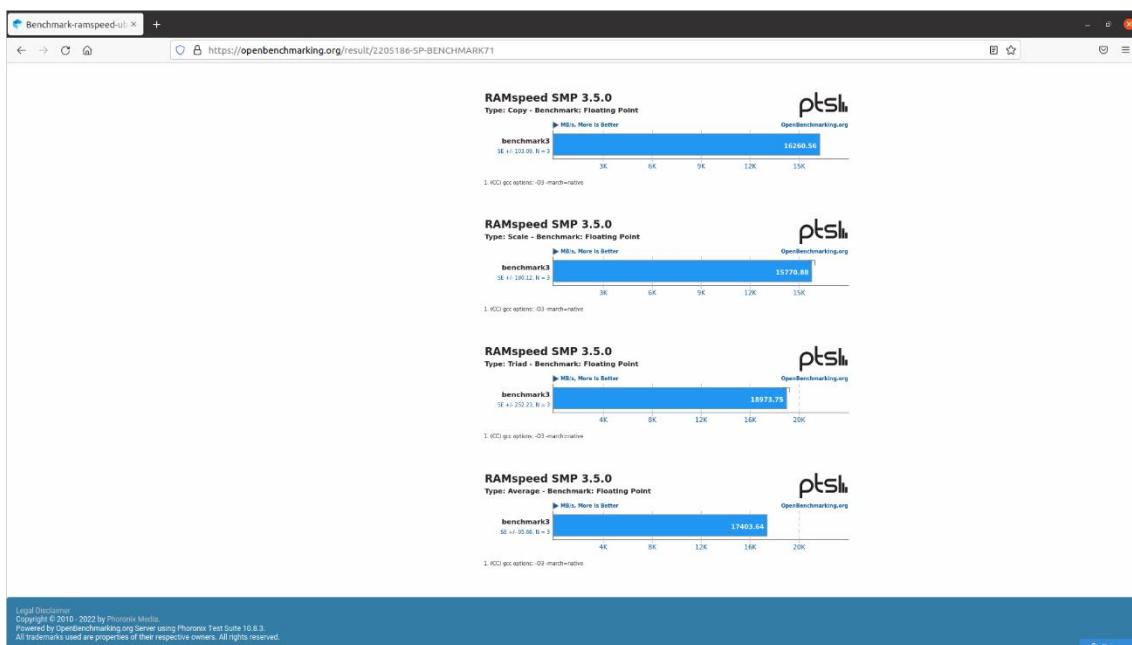
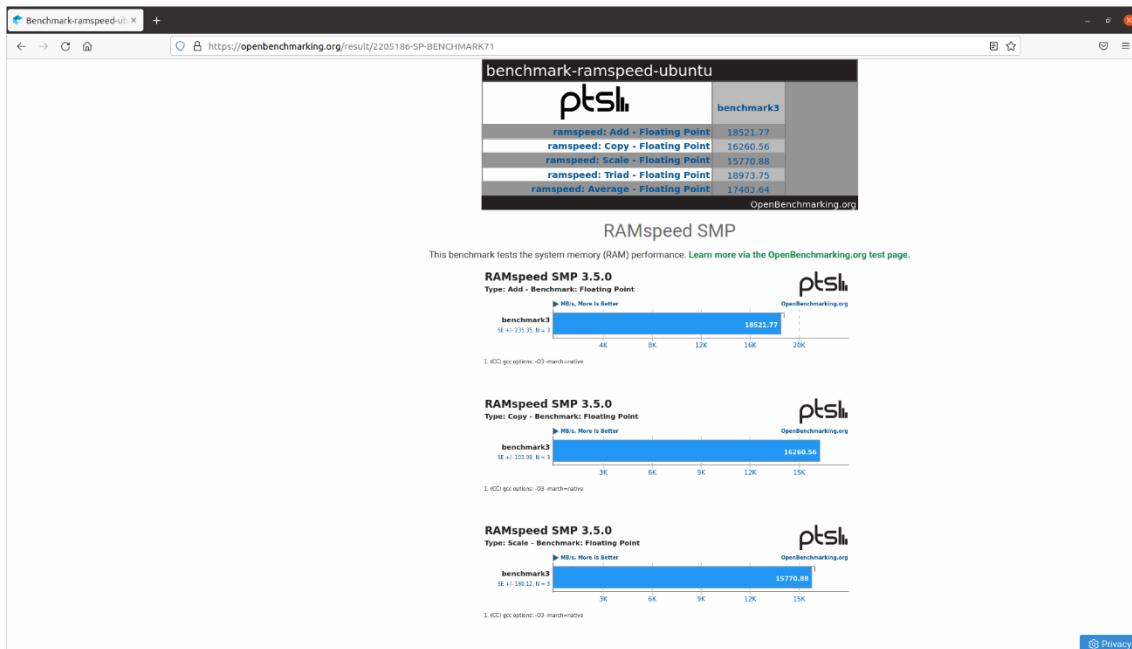
```

--build=x86_64-linux-gnu --disable-vtable-verify
--disable-werror --enable-checking=release
--enable-clccale=gnat --enable-default-pie
--enable-finite-objects
--enable-languages=c,ada,c++,go,brig,d,fortran,obj-c++,gn
--enable-libstdcxx-debug --enable-libstdcxx-time=yes
--enable-multilib --enable-multilib --enable-nls
--enable-objc-gc=auto
--enable-offload-targets=nvptx-none --build=gcc-9-Av3uEd/gcc-9-
--enable-plugin --enable-shared
--enable-threads=posix --host=x86_64-linux-gnu
--program-prefix=x86_64-linux-gnu-
--target=x86_64-linux-gnu --with-abi=m64
--with-arch-32=i686 --with-default-libstdcxx-abi=new
--with-gcc-major-version-only
--with-multilib-list=m32,m64,mx32
--with-target-system-zlib=auto --with-tune=generic
--without-cuda-driver -v

```

**Notes:**

- KPTI + usercopy/swaps barriers and \_\_user pointer sanitization + Retpolines STIBP: disabled RSB filling System Logs + PTE Inversion



En resumen, hemos obtenido los siguientes resultados tras ejecutar el benchmark RAMSpeed en Ubuntu Server:

Sistema Operativo	Ubuntu Server 20.04
<b>RAM</b>	5GB – 5120 MB
<b>Benchmark</b>	<i>RAMSpeed</i>
<b>Test 1: Add</b>	18521.77 MB/s
<b>Test 2: Copy</b>	16260.56 MB/s
<b>Test 3: Scale</b>	15770.88 MB/s
<b>Test 4: Triad</b>	18973.75 MB/s
<b>Test 5: Average</b>	17403.64 MB/s

## 1.4.2. Ejecución del Benchmark Sudokut en Ubuntu Server

Lanzamos el comando *phoronix-test-suite benchmark sudokut* y esperamos un poco a que el propio programa instale las dependencias necesarias para ejecutarse.

```
juanmaarg6@ise-ubuntu:~$ phoronix-test-suite benchmark sudokut

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 10830.
VISIT HTTPS://WWW.PHORONIX-TEST-SUITE.COM/ TO UPDATE THIS SOFTWARE.

The following dependencies are needed and will be installed:
- tcl

This process may take several minutes.
```

Tras ello, nos mostrará la siguiente salida:

```
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for libc-bin (2.31-dubuntu9.7) ...
To Install: pts/sudokut-1.0.i

Determining File Requirements .....  
Searching Download Caches .....
```

1 Test To Install  
1 File Needed [0.02 MB / 1 Minute]  
1MB Of Disk Space Is Needed

```
pts/sudokut-1.0.i  
-----  
File: sudokut-1.0.i  
1 File Needed [0.02 MB / 1 Minute]  
Downloading: sudokut-0.4.1.tar.bz2  
Estimated Download Time: 1m .....  
[0.02MB]  
Installation Size: 0.1 MB  
Installing Test @ 22:22:07
```

**System Information**

```
PROCESSOR: Intel Core i7-7700HQ
Core Count: 1
Extensions: SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE
Cache Size: 0144 KB

GRAPHICS: VMware SVGA II
Screen: 2048x2048

PERIPHERALS:
BIOS Version: VirtualBox
Chipset: Intel 440FX B2441FX PMC
Audio: Intel 8280IAA AC 97 Audio
Network: 2 x Intel 82540EM

MEMORY:
Disk: 2 x 11GB VBOX HD
File-System: ext4
Mount Options: relatime rw
Disk Scheduler: PQ_DEADLINE

OPERATING SYSTEM:
Kernel: Linux 4.15.0-102-generic (x86_64)
Compiler: GCC 9.4.0
System Layer: Oracle VMware
Security:
+ usercopy/swaps barriers and user pointer sanitization
+ heap polling SIBP: disabled HSA filling
+ PTE Inversion
```

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

Nos preguntará si queremos guardar los resultados en un fichero. Esta vez, diremos que no.

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

Tras ejecutarse el benchmark obtenemos el siguiente resultado:

```
Sudokut 0.4:  
pts/sudokut-1.0.1  
Test 1 of 1  
Estimated Trial Run Count: 3  
Estimated Time To Completion: 2 Minutes [22:24 UTC]  
    Started Run 1 @ 22:22:39  
    Started Run 2 @ 22:22:53  
    Started Run 3 @ 22:23:06  
  
Total Time:  
    13.314419984818  
    12.486961841583  
    12.739302158356  
  
Average: 12.85 Seconds  
Deviation: 3.30%  
  
juanmaarg6@ise-ubuntu:~$
```

Esta vez no nos pide si queremos ver los resultados almacenados en un archivo ya que antes confirmamos que no queríamos crear un archivo para almacenar los resultados. Por la misma razón, no nos pide si queremos subir nuestros resultados de ejecución a OpenBenchmarking.org.

En resumen, hemos obtenido los siguientes resultados tras ejecutar el benchmark Sudokut en Ubuntu Server:

<b>Sistema Operativo</b>	<i>Ubuntu Server 20.04</i>
<b>Benchmark</b>	<i>Sudokut</i>
<b>Tiempo medio resultante</b>	12.85 segundos
<b>Desviación típica</b>	3.30%

## **1.5. Quinto Paso: Comparativa de los Resultados de los Benchmarks**

### **1.5.1. Comparativa de los Resultados de Ejecución de RAMSpeed**

Sistema Operativo	<i>CentOS Linux 8</i>	<i>Ubuntu Server 20.04</i>
<b>RAM</b>	5GB – 5120 MB	5GB – 5120 MB
<b>Benchmark</b>	<i>RAMSpeed</i>	<i>RAMSpeed</i>
<b>Test 1: Add</b>	19118.68 MB/s	18521.77 MB/s
<b>Test 2: Copy</b>	16937.82 MB/s	16260.56 MB/s
<b>Test 3: Scale</b>	16685.43 MB/s	15770.88 MB/s
<b>Test 4: Triad</b>	18902.83 MB/s	18973.75 MB/s
<b>Test 5: Average</b>	17736.27 MB/s	17403.64 MB/s

Aunque no se observan grandes diferencias, podemos decir que obtenemos unos resultados un poco mejores (More is Better) en CentOS. Esto puede deberse a que en Ubuntu Server tenemos más servicios ejecutándose en segundo plano cuando iniciamos la máquina (servicios que hemos instalado en anteriores prácticas), de forma que la máquina es un poco más lenta y hace que CentOS sea un sistema más “ligero”.

### **1.5.2. Comparativa de los Resultados de Ejecución de Sudokut**

Sistema Operativo	<i>CentOS Linux 8</i>	<i>Ubuntu Server 20.04</i>
<b>Benchmark</b>	<i>Sudokut</i>	<i>Sudokut</i>
<b>Tiempo medio resultante</b>	13.07 segundos	12.85 segundos
<b>Desviación típica</b>	0.33%	3.30%

Al igual que antes, no se observan grandes diferencias en el tiempo medio resultante, aunque podemos decir que obtenemos unos resultados un poco mejores (en este caso, Fewer is Better) en Ubuntu Server. Sin embargo, si que notamos una diferente más significativa en la desviación típica, siendo esta mayor en Ubuntu Server.

## **2. Ejercicio 2**

### **2.0. Enunciado del Ejercicio**

*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 Expressión Extractor) para extraer el token JWT que hay que añadir a la cabecera de las peticiones (usando HTTP Header Manager).*

### **2.1. Primer Paso: Instalación de JMeter en nuestra Máquina Anfitrión**

Comenzamos instalando el programa JMeter, que será el entorno de desarrollo principal del ejercicio, en nuestra máquina anfitrión. En mi caso, estoy utilizando Ubuntu 20.04.

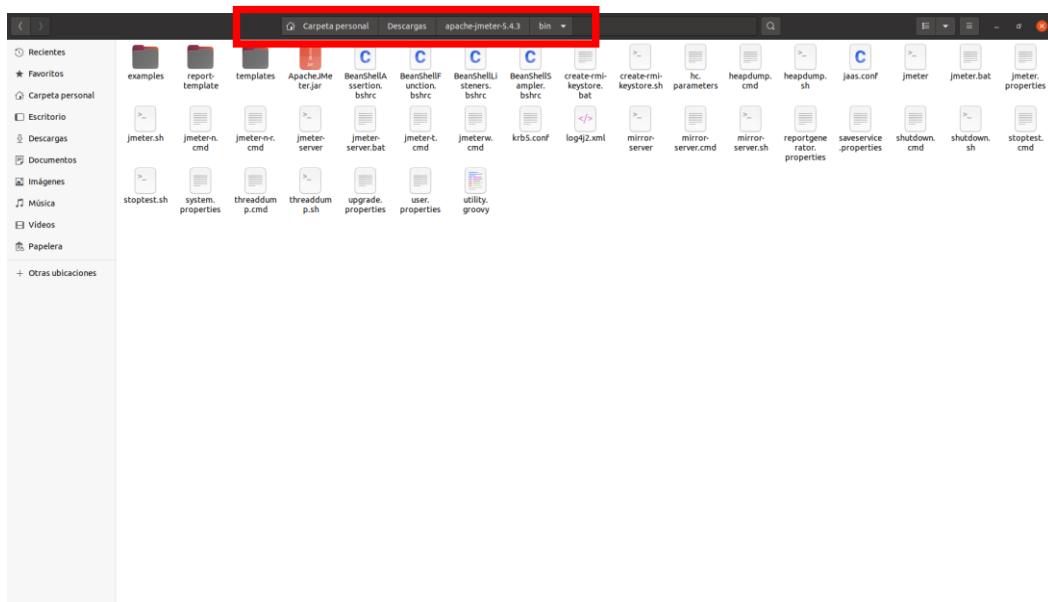
Podemos instalarlo desde el propio gestor de paquetes de Ubuntu con el comando `sudo apt-get install jmeter`, sin embargo, se nos instalará una versión antigua (versión 2.8), mientras que la versión actual es la 5.4.3, luego, es más recomendable instalarlo manualmente desde la página oficial de JMeter.

Nos vamos a la página oficial de JMeter y descargamos el binario correspondiente:

[https://jmeter.apache.org/download\\_jmeter.cgi](https://jmeter.apache.org/download_jmeter.cgi)

The screenshot shows the Apache JMeter download page. On the left, there's a sidebar with links for About, Download, Documentation, Tutorials, Community, and Foundation. The main content area has a header 'Download Apache JMeter'. Below it, a note says 'We recommend you use a mirror to download our release builds, but you must verify the integrity of the downloaded files using signatures downloaded from our main distribution directories. Recent releases (48 hours) may not yet be available from all the mirrors.' It also mentions 'Other mirrors' and 'KEYS'. The central part is titled 'Apache JMeter 5.4.3 (Requires Java 8+)' and shows the 'Binaries' section with two download links: 'apache-jmeter-5.4.3.tgz sha512.pgp' and 'apache-jmeter-5.4.3.zip sha512.pgp'. Below that is the 'Source' section with links for 'apache-jmeter-5.4.3\_src.tgz sha512.pgp' and 'apache-jmeter-5.4.3\_src.zip sha512.pgp'. At the bottom, there's an 'Archives' section.

Descomprimimos el archivo *tgz* que hemos descargado y vamos a acceder mediante la terminal a la siguiente carpeta del archivo:

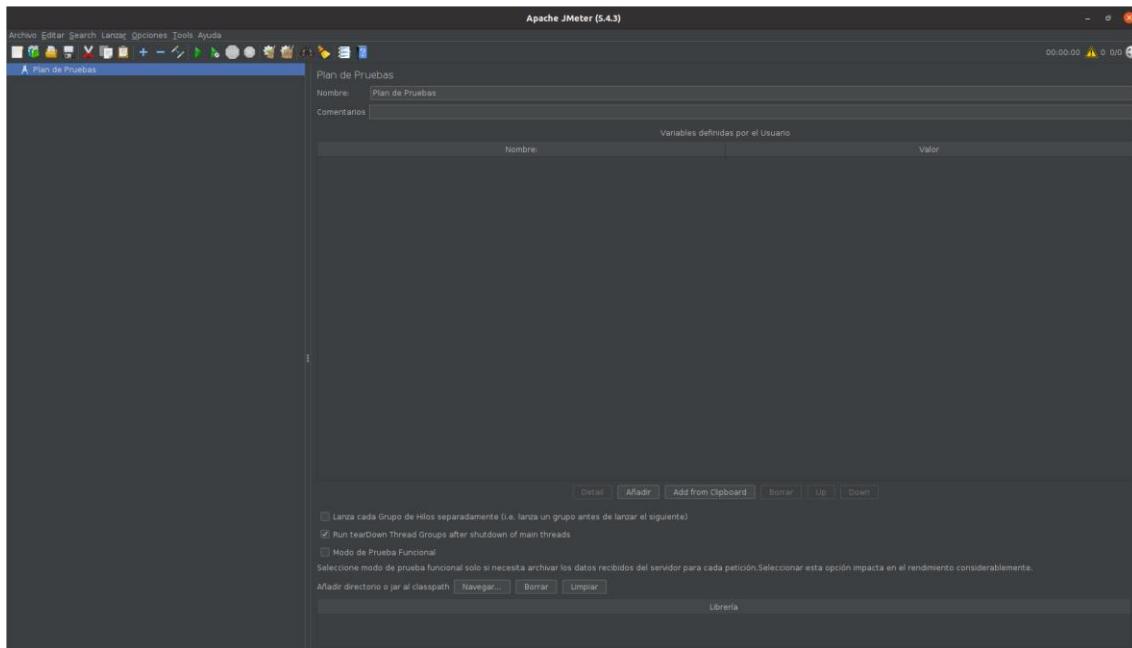


Una vez estemos situados en dicha carpeta en la terminal (lanzamos el comando `cd $HOME/Descargas/apache-jmeter-5.4.3/bin`), le damos permisos de ejecución al archivo *ApacheJMeter.jar* y ejecutamos JMeter con el comando `java -jar ApacheJMeter.jar`.

```
juanma@Lenovo-JuanmaLinux:~/Descargas/apache-jmeter-5.4.3/bin$ sudo chmod +x ApacheJMeter.jar
[sudo] contraseña para juanma:
juanma@Lenovo-JuanmaLinux:~/Descargas/apache-jmeter-5.4.3/bin$
```

```
juanma@Lenovo-JuanmaLinux:~/Descargas/apache-jmeter-5.4.3/bin$ java -jar ApacheJMeter.jar
=====
Don't use GUI mode for load testing !, only for Test creation and Test debugging.
For load testing, use CLI Mode (was NON GUI):
  jmeter -n -t [jmx file] -l [results file] -e -o [Path to web report folder]
& increase Java Heap to meet your test requirements:
  Modify current env variable HEAP="-Xms1g -Xmx1g -XX:MaxMetaspaceSize=256m" in the jmeter batch file
Check : https://jmeter.apache.org/usermanual/best-practices.html
=====
may. 25, 2022 2:31:15 P. M. com.kitfox.svg.Text buildText
ADVERTENCIA: Could not create font Arial
may. 25, 2022 2:31:15 P. M. com.kitfox.svg.Text buildText
```

Luego se nos debe de abrir JMeter, el cual tiene el siguiente aspecto:



## 2.2. Segundo Paso: Instalación de Docker en Ubuntu Server

Vamos a instalar Docker y Docker-Compose en Ubuntu Server. Para ello, seguimos los pasos indicados en el guión de la práctica. Puesto que no podemos copiar y pegar comandos desde nuestro localhost a VirtualBox, vamos a conectarnos por SSH desde nuestra máquina anfitrión a la máquina de Ubuntu Server para así copiar y pegar los comandos y evitar errores.

```
juanma@Lenovo-JuanmaLinux:~$ ssh juanmaarg6@192.168.56.105 -p 22022
juanmaarg6@192.168.56.105's password:
Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.4.0-109-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

System information disabled due to load higher than 1.0

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation

8 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Last login: Thu May 19 15:50:10 2022
juanmaarg6@ise-ubuntu:~$
```

Añadimos la clave GPG para validar el repositorio de Docker:

```
juanmaarg6@ise-ubuntu:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
[sudo] password for juanmaarg6:
OK
juanmaarg6@ise-ubuntu:~$
```

Añadimos el repositorio:

```
juanmaarg6@ise-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
Hit:2 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Hit:3 http://repo.zabbix.com/zabbix/5.0/ubuntu focal InRelease
Get:4 https://download.docker.com/linux/ubuntu focal InRelease [57.7 kB]
Get:5 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:6 http://es.archive.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:7 http://es.archive.ubuntu.com/ubuntu focal amd64 Contents (deb) [40.9 MB]
Get:8 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [16.7 kB]
Get:9 https://download.docker.com/linux/ubuntu focal/stable amd64 Contents (deb) [1361 B]
Get:10 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1793 kB]
Get:11 http://es.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [330 kB]
Get:12 http://es.archive.ubuntu.com/ubuntu focal-updates amd64 Contents (deb) [116 MB]
Get:13 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [15.2 kB]
Get:14 http://es.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [976 kB]
Get:15 http://es.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [139 kB]
```

Ahora actualizamos los repositorios de Ubuntu Server:

```
juanmaarg6@ise-ubuntu:~$ sudo apt update
Hit:1 http://es.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 https://download.docker.com/linux/ubuntu focal InRelease
Hit:3 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://repo.zabbix.com/zabbix/5.0/ubuntu focal InRelease
Hit:5 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:6 http://es.archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
35 packages can be upgraded. Run 'apt list --upgradable' to see them.
juanmaarg6@ise-ubuntu:~$
```

Buscamos en el gestor el paquete docker-ce (Community Edition) y lo instalamos:

```
juanmaarg6@ise-ubuntu:~$ apt search docker-ce
Sorting... Done
Full Text Search... Done
docker-ce/focal 5:20.10.16~3-0~ubuntu-focal amd64
  Docker: the open-source application container engine

docker-ce-cli/focal 5:20.10.16~3-0~ubuntu-focal amd64
  Docker CLI: the open-source application container engine

docker-ce-rootless-extras/focal 5:20.10.16~3-0~ubuntu-focal amd64
  Rootless support for Docker.

juanmaarg6@ise-ubuntu:~$
```

```
juanmaarg6@ise-ubuntu:~$ sudo apt install docker-ce
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
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 cgroupsfs-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 35 not upgraded.
Need to get 102 MB of archives.
After this operation, 422 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 https://download.docker.com/linux/ubuntu focal/stable amd64 containerd.io amd64 1.6.4-1 [28.1 MB]
Get:2 http://es.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57.4 kB]
Get:3 http://es.archive.ubuntu.com/ubuntu focal/universe amd64 slirp4netns amd64 0.4.3-1 [74.3 kB]
Get:4 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce-cli amd64 5:20.10.16~3-0~ubu
Get:5 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce amd64 5:20.10.16~3-0~ubu
```

Comprobamos el estado del servicio Docker y en el caso de que esté inactivo, lo iniciamos.

```
juanmaarg6@lse-ubuntu:~$ 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
juanmaarg6@lse-ubuntu:~$ systemctl start docker
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ===
Authentication is required to start 'docker.service'.
Authenticating as: lse-ubuntu (juanmaarg6)
Password:
==== AUTHENTICATION COMPLETE ===
juanmaarg6@lse-ubuntu:~$ systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2022-05-19 16:09:20 UTC; 2s ago
    TriggeredBy: ● docker.socket
      Docs: https://docs.docker.com
      Main PID: 5052 (dockerd)
        Tasks: 7
       Memory: 29.3M
      CGroup: /system.slice/docker.service
              └─5052 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

May 19 16:09:16 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:16.216895859Z" level=warning msg="You
May 19 16:09:16 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:16.217351447Z" level=warning msg="You
May 19 16:09:16 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:16.217696261Z" level=warning msg="You
May 19 16:09:16 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:16.218500087Z" level=info msg="Loadin
May 19 16:09:17 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:17.747800296Z" level=info msg="Defaul
May 19 16:09:18 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:18.501375640Z" level=info msg="Loadin
May 19 16:09:19 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:19.429430740Z" level=info msg="Docker
May 19 16:09:19 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:19.431446137Z" level=info msg="Daemon
May 19 16:09:20 lse-ubuntu systemd[1]: Started Docker Application Container Engine.
May 19 16:09:20 lse-ubuntu dockerd[5052]: time="2022-05-19T16:09:20.184299597Z" level=info msg="API li
lines 1-21/21 (END)
```

Ahora añadimos nuestro usuario al grupo docker:

```
juanmaarg6@lse-ubuntu:~$ sudo usermod -aG docker juanmaarg6
juanmaarg6@lse-ubuntu:~$ groups juanmaarg6
juanmaarg6 : juanmaarg6 adm cdrom sudo dip plugdev lxd docker
juanmaarg6@lse-ubuntu:~$
```

Comprobamos la información de Docker y ejecutamos un script de prueba:

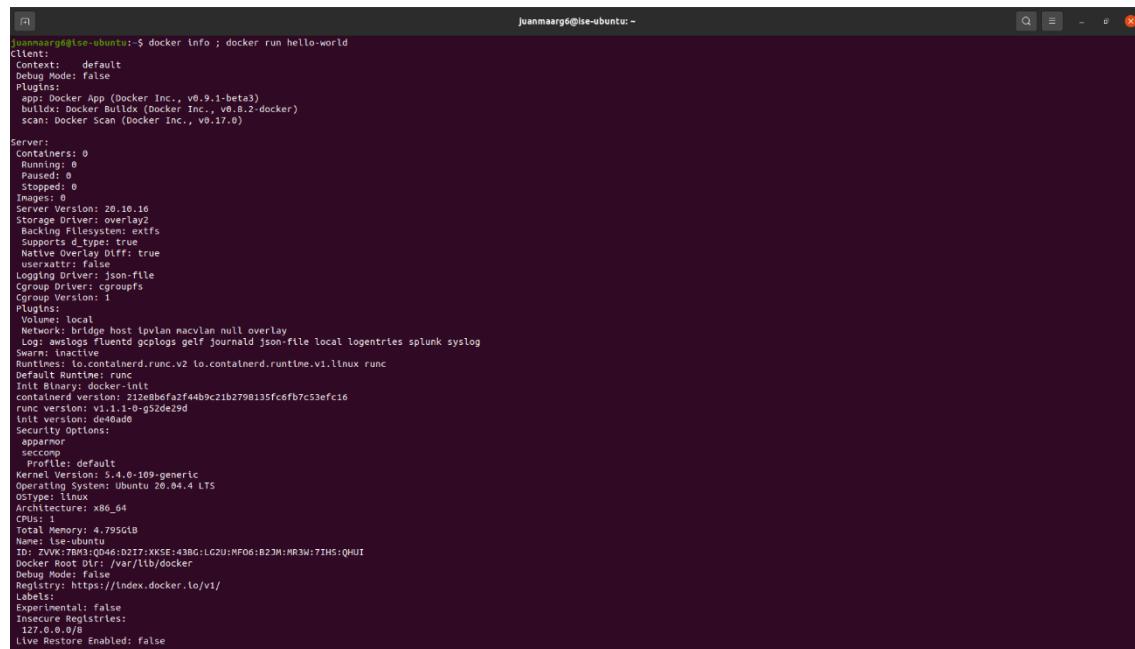
```
juanmaarg6@lse-ubuntu:~$ docker info
Client:
  Context: default
  Debug Mode: false
  Plugins:
    app: Docker App (Docker Inc., v0.9.1-beta3)
    buildx: Docker Buildx (Docker Inc., v0.8.2-docker)
    scan: Docker Scan (Docker Inc., v0.17.0)

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/info": dial unix /var/run/docker.sock: connection refused
  See 'docker run --help'.
juanmaarg6@lse-ubuntu:~$
```

Vemos que nos da error. Al buscar la solución al error, encontré lo siguiente:

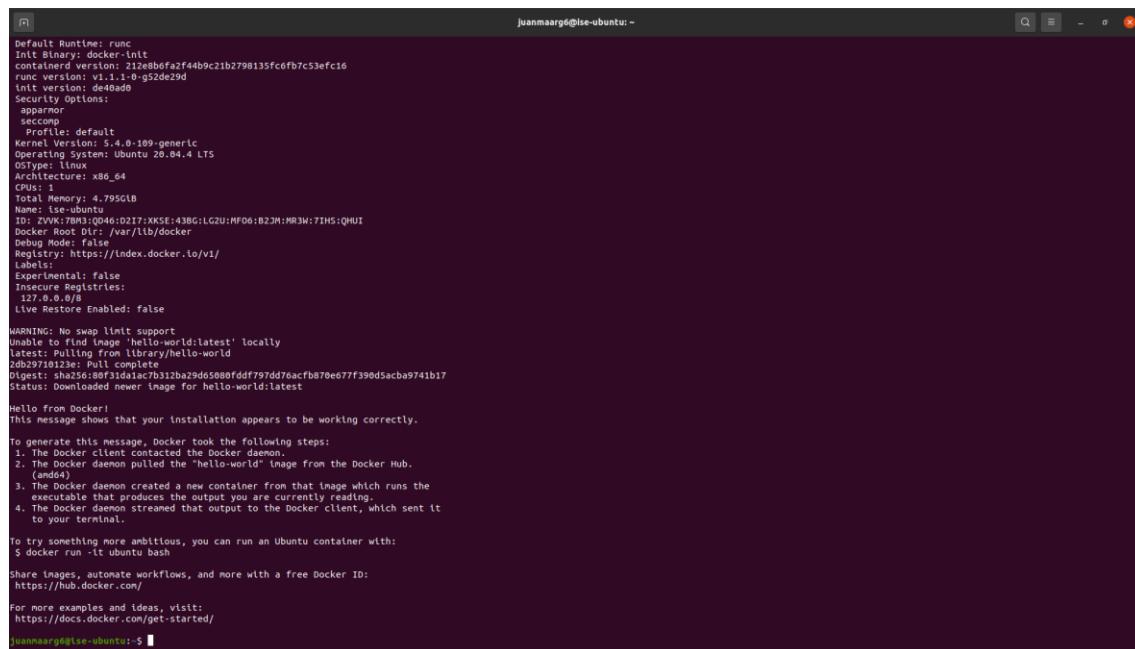
<https://www.drupaladicto.com/snippet/como-corregir-error-docker-got-permission-denied-while-trying-connect-docker-daemon-socket>

En resumen, tenemos que reiniciar nuestra sesión de usuario con el comando `su – juanmaarg6` en mi caso o, alternativamente, reiniciar el sistema. Una vez hecho esto, al comprobar la información de Docker e intentar ejecutar el script de prueba, vemos que todo se realiza correctamente.



```
juanmaarg6@lse-ubuntu:~$ docker info ; docker run hello-world
Client:
  Context:    default
  Debug Mode: false
  Plugins:
    buildx (Docker App (Docker Inc., v0.9.1-beta3)
             buildx: Docker Buildx (Docker Inc., v0.8.2 docker)
             scan: Docker Scan (Docker Inc., v0.17.0))

Server:
  Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
  Total: 0
  Server Version: 20.10.10
  Storage Driver: overlay2
    Backing Filesystem: extfs
    Supports d_type: true
    Native Overlay Diff: true
  userattr: false
  Logging Driver: json-file
  Cgroup Driver: cgroupfs
  Container Version: 1
  Plugins:
    Volume: local
      Network: bridge host ipvlan macvlan null overlay
    Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
  Consoles: 0
  Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
  Default Runtime: runc
  Init Binary: docker-init
  containerd version: 212e8b6fa2f44b9c21b2798135fc0fb7c53efc16
  containerd commit: v1.1.1-0-g52de29d
  init version: de40ad0
  Security Options:
    apparmor
    seccomp
      Profile: default
    Kernel Version: 5.4.0-109-generic
    Operating System: Ubuntu 20.04.4 LTS
    OSType: linux
    Architecture: x86_64
    CPUs: 1
    Total Memory: 4.795GiB
  Name: lse-ubuntu
  ID: TQVK7RM3Q046:DZI7:XKSE:43BG:LZU:MF06:BZJM:MR3W:7IH5:QHUI
  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
```



```
juanmaarg6@lse-ubuntu:~$ docker run hello-world
Default Runtime: runc
Init Binary: docker-init
Container ID: 212e8b6fa2f44b9c21b2798135fc0fb7c53efc16
Image: hello-world:latest
Image ID: sha256:80f31da1ac7b312ba29d65988ffdd797dd76acf8b70e077f390d5acba9741b17
Insecure Registries:
  127.0.0.0/8
Live Restore Enabled: false
WARNING: No swap limit support
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2d297f1023e: Pull complete
Digest: sha256:80f31da1ac7b312ba29d65988ffdd797dd76acf8b70e077f390d5acba9741b17
Status: Downloaded newer image for hello-world:latest

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.
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

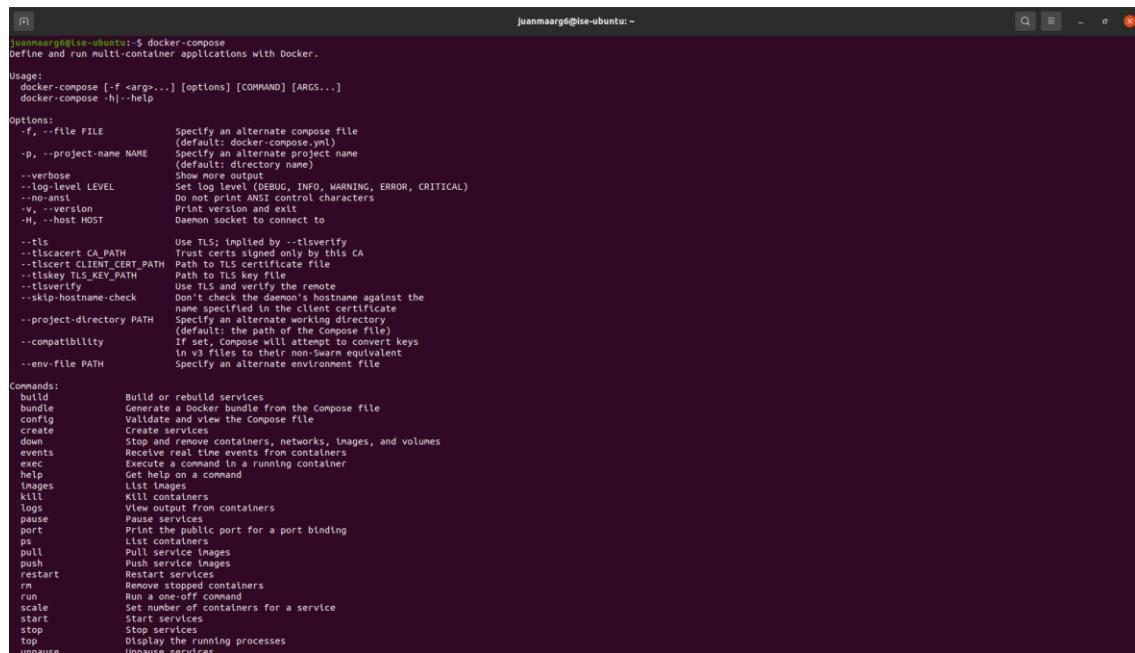
Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
juanmaarg6@lse-ubuntu:~$
```

Ahora procedemos a instalar docker-compose:

```
juanmaarg6@ise-ubuntu:~$ sudo apt install docker-compose
[sudo] password for juanmaarg6:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfwupdplugin1
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  python3-cached-property python3-docker python3-dockerpty python3-docopt python3-pkg-resources
Recommended packages:
  docker.io
The following NEW packages will be installed:
  docker-compose python3-cached-property python3-docker python3-dockerpty python3-pkg-resources
0 upgraded, 7 newly installed, 0 to remove and 35 not upgraded.
Need to get 262 kB of archives.
After this operation, 1616 kB of additional disk space will be used.
Do you want to continue? [Y/n] ■
```

Finalmente, probamos a lanzar a ejecutar el comando *docker-compose* y a comprobar la versión instalada, y vemos que todo se ejecuta perfectamente.



```
juanmaarg6@ise-ubuntu:~$ docker-compose
Define and run multi-container applications with Docker.

Usage:
  docker-compose [-f <arg>...] [options] [COMMAND] [ARGS...]
  docker-compose -h|--help

Options:
  -f, --file FILE          Specify an alternate compose file
  -p, --project-name NAME   Specify an alternate project name
                            (default: docker-compose.yml)
  --verbose                 Show more output
  --log-level LEVEL         Set log level (DEBUG, INFO, WARNING, ERROR, CRITICAL)
  --no-color                Do not use colors and control characters
  -v, --version              Print version and exit
  -H, --host HOST           Daemon socket to connect to

  --tls                      Use TLS; implied by --tlsverify
  --tlscacert CA_PATH        Trust certs signed only by this CA
  --tlscert CLIENT_CERT_PATH Path to TLS certificate file
  --tlskey TLS_KEY_PATH      Path to TLS key file
  --tlsverify                Use TLS and verify the remote
  --skip-hostname-check      Don't check the host's hostname against the
                            name specified in the client certificate
  --project-directory PATH   Specify an alternate working directory
                            (default: the path of the compose file)
  --compatibility            If set, Compose will attempt to convert keys
                            in .vs files to their non-Swarm equivalent
  --env-file PATH            Specify an alternate environment file

Commands:
  build      Build or rebuild services
  bundle     Generate a Docker bundle from the Compose file
  config     Validate and view the Compose file
  create     Create services
  down       Stop and remove containers, networks, images, and volumes
  events     Receive real-time events from containers
  exec       Execute a command in a running container
  help       Get help on a command
  images    List images
  kill       Kill containers
  logs      View output from containers
  pause     Pause services
  port      Print the public port for a port binding
  ps        List containers
  pull      Pull service images
  push      Push service images
  restart   Restart services
  rm        Remove stopped containers
  run       Run a one-off command
  scale    Set number of containers for a service
  start    Start services
  stop     Stop services
  top       Display the running processes
  unname   Unname services
```

```
juanmaarg6@ise-ubuntu:~$ docker-compose --version
docker-compose version 1.25.0, build unknown
juanmaarg6@ise-ubuntu:~$ ■
```

## **2.3. Tercer Paso: Instalación de la Aplicación iseP4JMeter en Ubuntu Server**

Clonamos el repositorio de Git dado en la máquina de Ubuntu Server (dicho repositorio contiene la aplicación iseP4JMeter).

```
juanmaarg6@ise-ubuntu:~$ git clone https://github.com/davidPalomar-ugr/iseP4JMeter.git
Cloning into 'iseP4JMeter'...
remote: Enumerating objects: 3801, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 3801 (delta 12), reused 15 (delta 7), pack-reused 3774
Receiving objects: 100% (3801/3801), 7.80 MiB | 3.64 MiB/s, done.
Resolving deltas: 100% (718/718), done.
juanmaarg6@ise-ubuntu:~$
```

Tras esto, tendremos en nuestra máquina de Ubuntu Server, un nuevo directorio, *iseP4Jmeter*. Nos metemos en él y podemos levantar la aplicación con el comando *docker-compose up*.

```
juanmaarg6@ise-ubuntu:~$ cd iseP4JMeter
juanmaarg6@ise-ubuntu:~/iseP4JMeter$ docker-compose up
Creating network "isepjmeter_default" with the default driver
Pulling mongodb (mongo:...):
latest: Pulling from library/mongo
5sf01ec1767: Pulling fs layer
52c91407ff49: Downloading [=====
05bd3555a96c: Downloading [>-----]
05bd3555a96c: Downloading [=====] b2c91407ff49: Download complete
d5fd17ec1767: Pull complete
05bd3555a96c: Pull complete
05bd3555a96c: Pull complete
49f0e29b2e16: Pull complete
01c07089c846: Pull complete
0c74ee88dd5a: Pull complete
8aecb2f59c4d: Pull complete
6a1ab21c2a9: Pull complete
e53674bec5a: Pull complete
Digest: sha256:ebdbdd0997087ff398087d4e064218d477e9611a7becd78664a2ab490eff
Status: Downloaded newer image for mongo:latest
Building mongodinit
Step 1/5 : FROM mongo
--> 27dd1be4bed1
Step 2/5 : COPY ./scripts/* /tmp/
--> 0abafaf74353
Step 3/5 : RUN /tmp/initMongoDB.sh
--> Running in aieb7b74cb37
Removing intermediate container aieb7b74cb37
--> 9ebad987f38
Step 4/5 : WORKDIR /tmp
--> Running in 0227382e6c16
Removing intermediate container 0227382e6c16
--> fde99d92e749
Step 5/5 : CMD ./InitializeMongoDB.sh
--> Running in 8e9e57e185b7
Removing intermediate container 8e9e57e185b7
--> 163cfbd2e015
Successfully built 163cfbd2e015
Successfully tagged isepjmeter_mongodinit:latest
WARNING: Image for service mongodinit was built because it did not already exist. To rebuild this image you must use 'docker-compose build' or 'docker-compose up --build'.
Building nodejs
Step 1/8 : FROM node:16.13.0-stretch
16.13.0-stretch: Pulling from library/node
200cae5cd98: Downloading [=====] 34.44MB/45.38MB
0172a2a25300: Downloading [=====] 1.14MB/1.14MB
e3cf3f384dd51: Download complete
cde69db633dd: Downloading [=====] 34.5MB/49.76MB
16c1fa47fbff: Downloading [=====] 25.25MB/214.4MB
fdf15e480cce: Waiting
a792262e0c5e: Waiting
452c56414df1: Waiting
e36ca80016b1: Waiting

```

```

juanmaarg6@ise-ubuntu:~/iseP4JMeter$ docker-compose up -d
iseP4JMeter_mongodb_1 is up-to-date
Starting isep4jmeter_mongodinit_1 ...
Starting isep4jmeter_mongodinit_1 ... done
juanmaarg6@ise-ubuntu:~/iseP4JMeter$ sudo ufw allow 3000/tcp
[sudo] password for juanmaarg6:
Rules updated
Rules updated (v6)
juanmaarg6@ise-ubuntu:~/iseP4JMeter$ █

```

En pantalla tendremos mostradas las peticiones que vayamos haciendo, podemos dejar docker como demonio o background con la opción -d.

```

juanmaarg6@ise-ubuntu:~/iseP4JMeter$ docker-compose up -d
iseP4JMeter_mongodb_1 is up-to-date
Starting isep4jmeter_mongodinit_1 ...
Starting isep4jmeter_mongodinit_1 ... done
juanmaarg6@ise-ubuntu:~/iseP4JMeter$ sudo ufw allow 3000/tcp
[sudo] password for juanmaarg6:
Rules updated
Rules updated (v6)
juanmaarg6@ise-ubuntu:~/iseP4JMeter$ █

```

Como se puede observar en la imagen anterior, en caso de que tengamos el firewall de Ubuntu Server, *ufw*, activo, tenemos que permitir el tráfico en el puerto 3000 ya que dicho puerto lo usaremos en este ejercicio. Para ello, tenemos que lanzar el comando *sudo ufw allow 3000/tcp*.

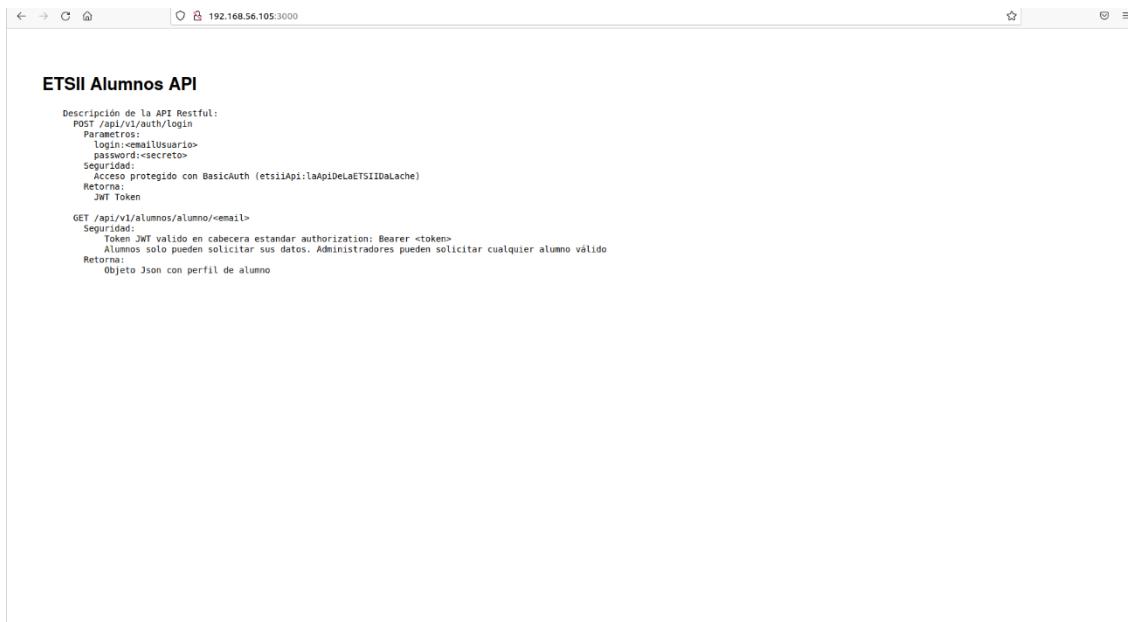
Si probamos a ejecutar el script *pruebaEntorno.sh* que se encuentra dentro del directorio *iseP4JMeter*, tendremos que obtener lo siguiente si todo está funcionando correctamente:

```

juanmaarg6@ise-ubuntu:~/iseP4JMeter$ ./pruebaEntorno.sh
{"id": "628b5217fdca7e7b97d3c42", "nombre": "Mari", "apellidos": "Fletcher Weiss", "sexo": "female", "email": "mariweiss@tropoli.com", "fechaNacimiento": "1992-04-04T00:00:00Z", "comentarios": "Aliquip dolor laboris ullamco id ex labore. Ipsum eiusmod ut aliquip non cillum deserunt sunt commodo anim ad nisi excepteur eu deserunt. Sint sunt proident Lorem irure irure minim adipisicing cillum. Nostrud officia in proident velit velit sit 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 aliqua. \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": "628b52211a274778db2c7441", "curso": 1, "media": 5.2}, {"id": "628b52211a274778db2c7442", "curso": 2, "media": 9.1}], "usuario": 10} juanmaarg6@ise-ubuntu:~/iseP4JMeter$ █

```

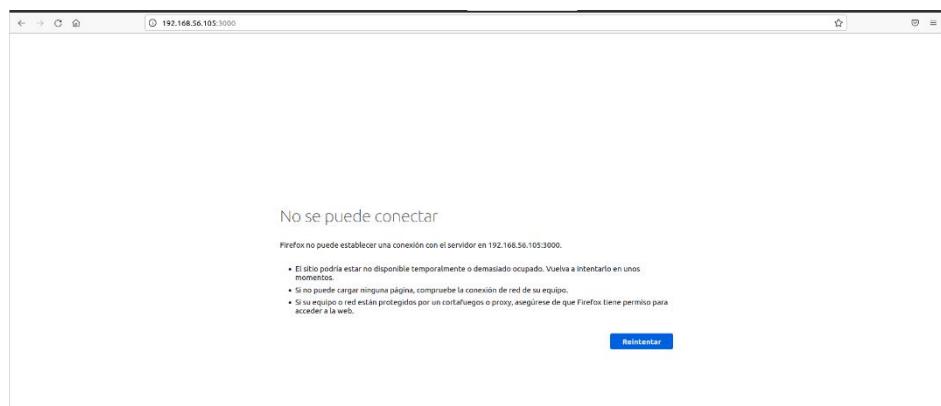
Por otro lado, si accedemos al navegador de nuestra máquina anfitrión y accedemos a la url <http://192.168.56.105:3000>, nos tiene que aparecer lo siguiente al tener la aplicación iseP4JMeter levantada:



Para hacer que la aplicación iseP4JMeter deje de funcionar, basta con ejecutar el comando `docker-compose down`.

```
juanmaarg6@ise-ubuntu:~/iseP4JMeter$ docker-compose down
Stopping isep4jmeter_nodejs_1 ... done
Stopping isep4jmeter_mongodb_1 ... done
Removing isep4jmeter_nodejs_1 ... done
Removing isep4jmeter_mongodbinit_1 ... done
Removing isep4jmeter_mongodb_1 ... done
Removing network isep4jmeter_default
juanmaarg6@ise-ubuntu:~/iseP4JMeter$
```

Si ahora accedemos como antes a la url <http://192.168.56.105:3000> con la aplicación iseP4JMeter no funcionando, obtenemos lo siguiente:



## **2.4. Cuarto Paso: Creación del Test en JMeter para la Aplicación iseP4JMeter**

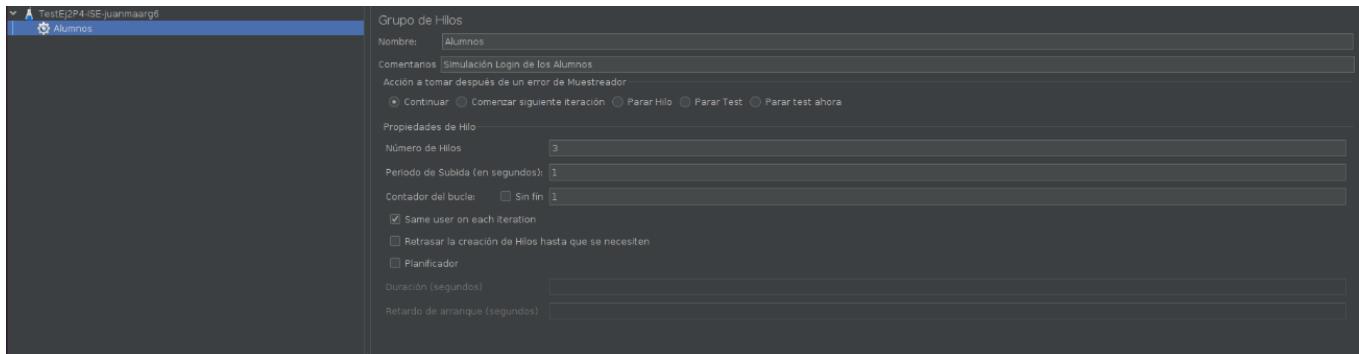
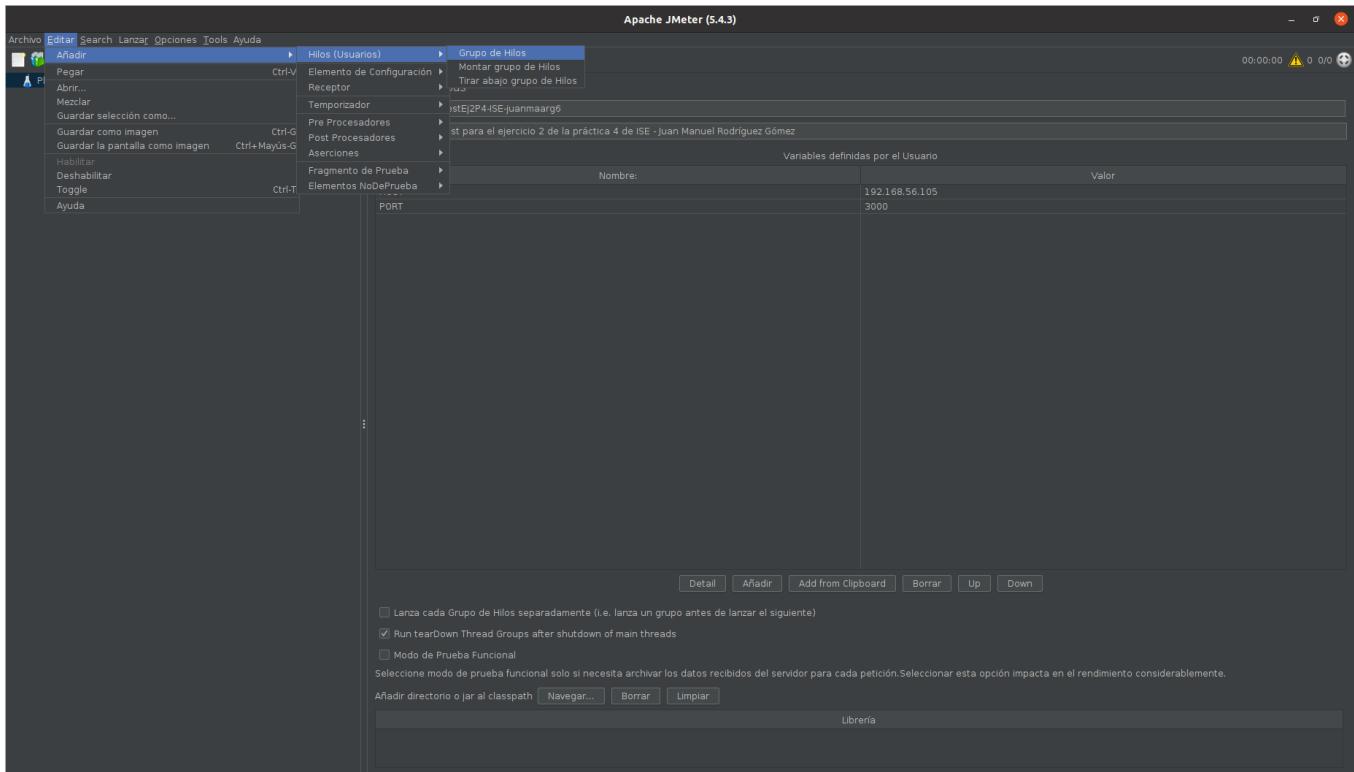
Vamos a indicar los pasos necesarios para crear en JMeter el test pedido para la aplicación iseP4JMeter. Cabe decir **que también necesitamos clonar el repositorio de Git dado en nuestra máquina anfitrión** (dicho repositorio contiene la aplicación iseP4JMeter).

### **1) Parametrización de HOST y PUERTO.**

The screenshot shows the JMeter Test Plan configuration window. At the top, there are fields for 'Nombre:' (Name) containing 'TestEj2P4-ISE-juanmaarg6' and 'Comentarios' (Comments) containing 'Test para el ejercicio 2 de la práctica 4 de ISE - Juan Manuel Rodríguez Gómez'. Below these, a table titled 'Variables definidas por el Usuario' (User-defined Variables) lists two variables: 'HOST' with value '192.168.56.105' and 'PORT' with value '3000'. At the bottom of the window are buttons for 'Detail', 'Añadir' (Add), 'Add from Clipboard', 'Borrar' (Delete), 'Up', and 'Down'.

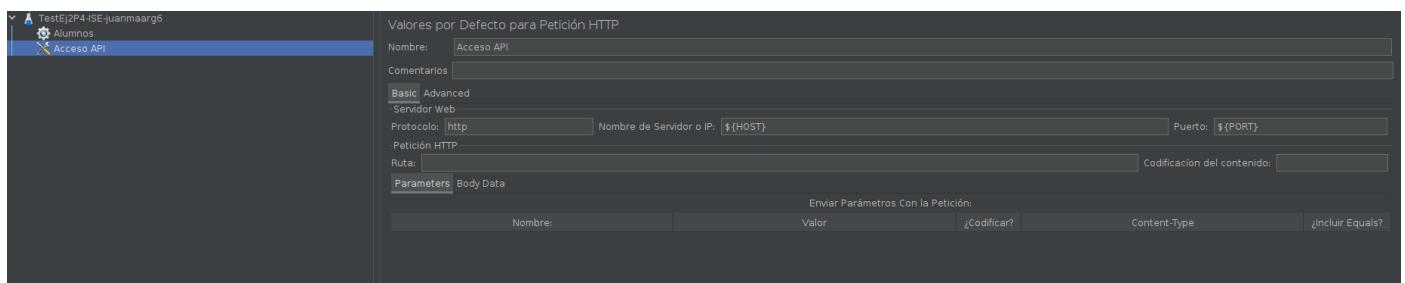
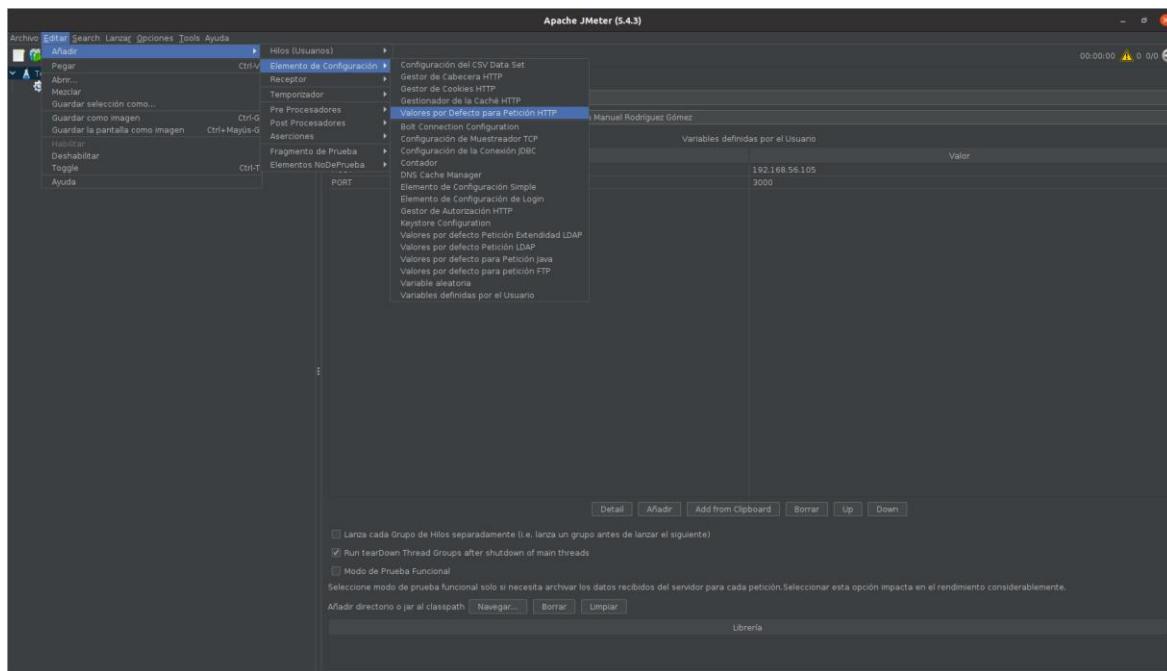
**2) Creación del grupo de hebras para los alumnos** (en mi caso, le he asignado 3 hebras a dicho grupo).

Para crear un grupo de hebras, nos situamos en **Plan de Pruebas** y luego: **Editar -> Añadir -> Hilos (Usuarios) -> Grupo de Hilos**.

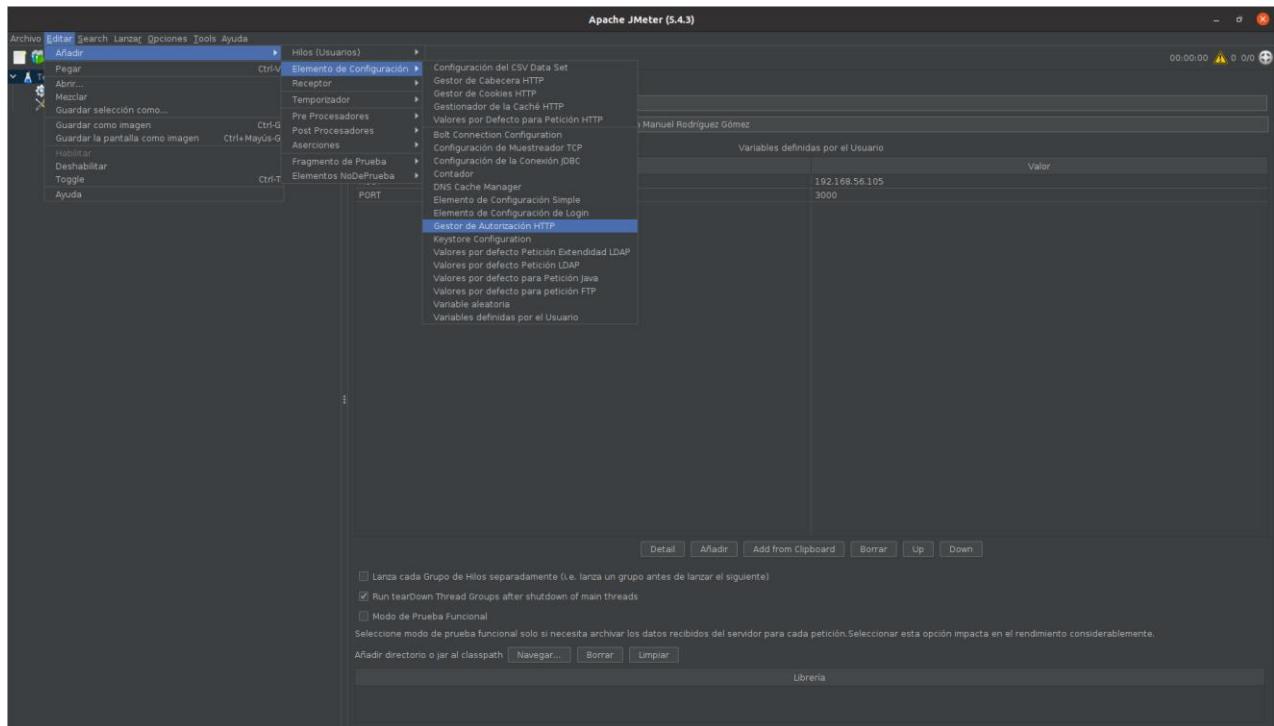


3) Añadir la petición de acceso al servidor. Como tenemos el host y el puerto parametrizados, basta con hacer referencia a esos valores.

Para ello, nos situamos en **Plan de Pruebas** y luego: **Editar -> Añadir -> Elemento de Configuración -> Valores por Defecto para Petición HTTP**.



- 4) Definir la autorización básica a la API.** Usamos los datos proporcionados en la página de GitHub: La URL Base sería `http://${HOST}:${PORT}/api/v1/auth/login`, el Nombre de Usuario es `etsiiApi` y la contraseña es `laApiDeLaETSIIDaLache`.  
 Nos situamos en **Plan de Pruebas** y luego: **Editar -> Añadir -> Elemento de Configuración -> Gestor de Autorización HTTP**.



TestEj2P4-ISE-juanmaarg

- Alumnos
- Acceso API
- Autorización Básica API

Gestor de Autorización HTTP

Nombre: Autorización Básica API

Comentarios

Options

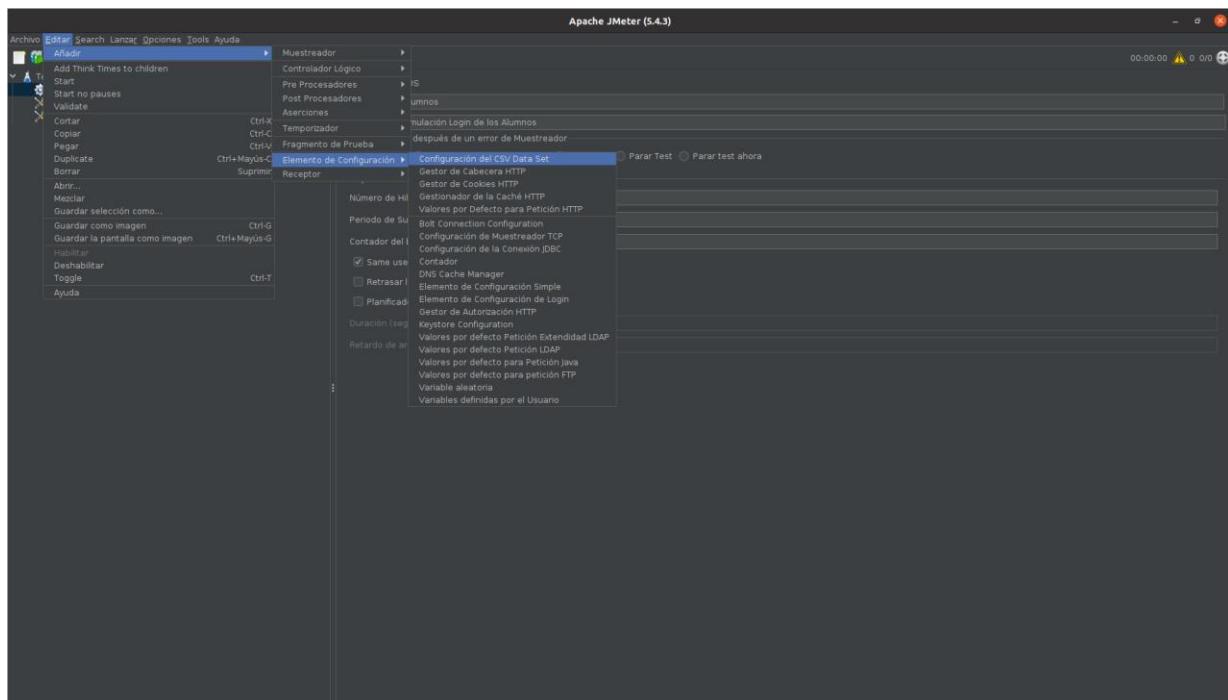
Clear auth on each iteration?

Use Thread Group configuration to control clearing

Autorizaciones Almacenadas en el Gestor de Autorización

URL Base	Nombre de Usuario	Contraseña	Dominio	Dominio (realm)	Mechanism
http://\${HOST}:\${PORT}/api/v1...	etsiiApi	.....			BASIC

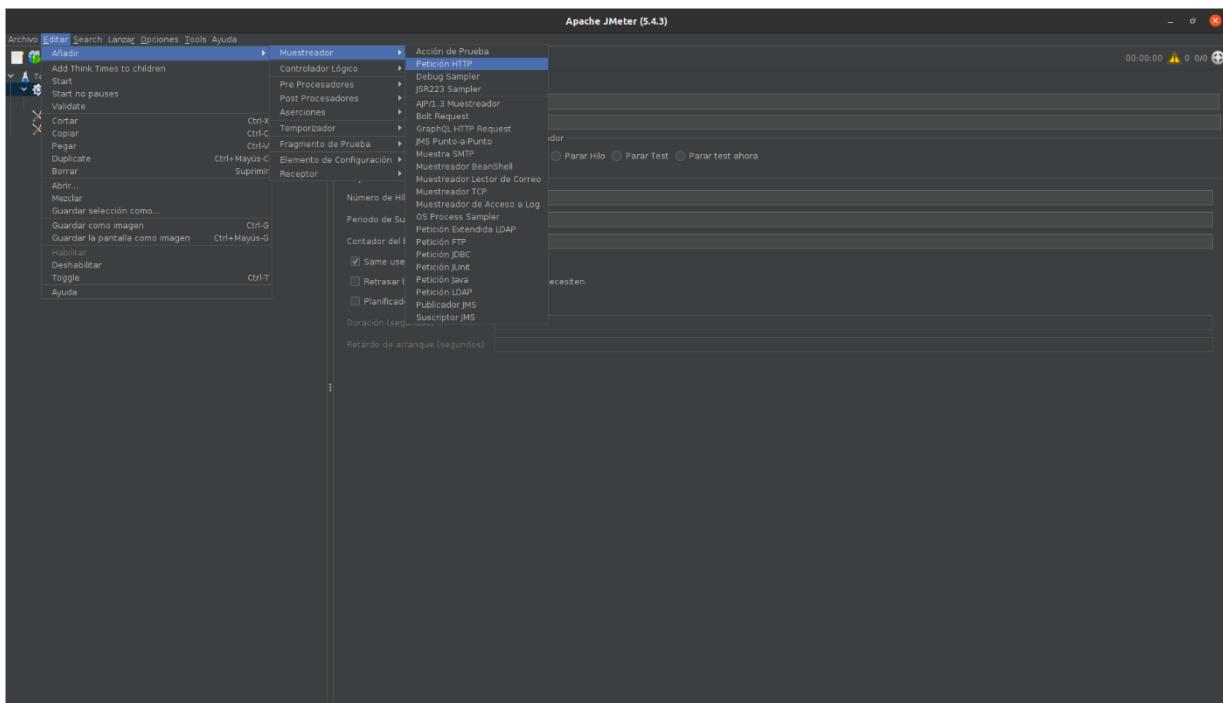
- 5) **Definir las credenciales para los Alumnos.** Le indicamos que dichas credenciales se encuentran en el fichero **alumnos.csv**, cuyos campos son usuario, contraseña.  
 Nos situamos en **Alumnos** y luego: **Editar -> Añadir -> Elemento de Configuración -> Configuración del CSV Data Set.**



Configuración del CSV Data Set
Nombre: Credenciales Alumnos
Comentarios:
Configura el Data Source de CSV
Nombre de Archivo: /home/juanma/Escritorio/iseP4JMeter/JMeter/alumnos.csv Navegar...
Codificación del fichero:
Nombres de Variable (delimitados por coma): login,password
Utilice la primera linea como Nombres de Variable: True
Delimitador (utilice ''\t'' para poner un tabulador): ,
Permitir datos entrecomillados: False
Reciclar en el fin de archivo (EOF): True
Para el hilo al final del fichero (EOF): False
Modo compartido: Actual grupo de hilos

## 6) Crear una petición POST para los Alumnos.

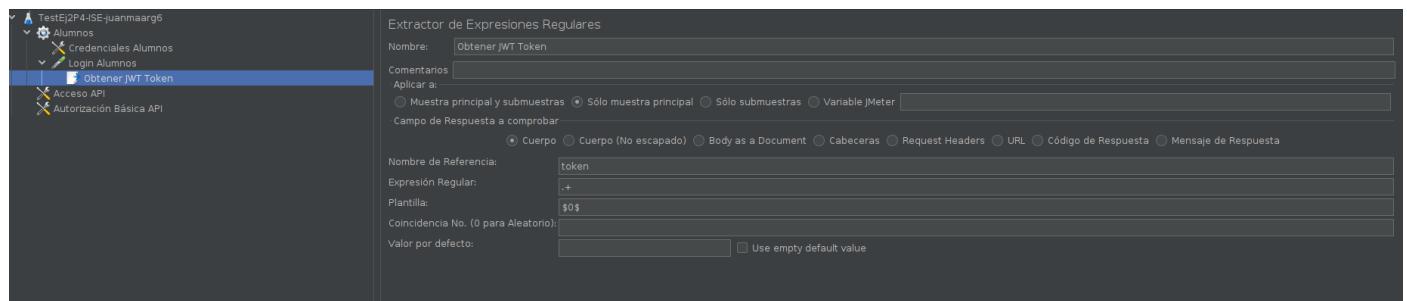
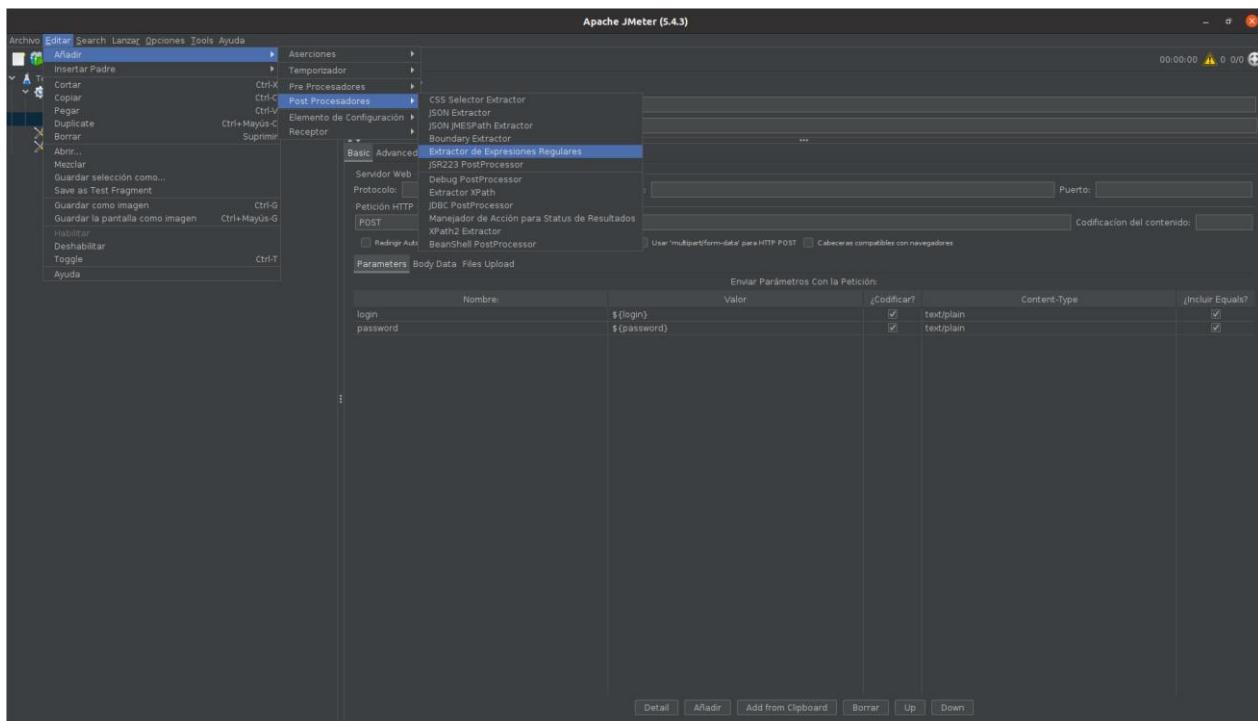
Nos situamos en **Alumnos** y luego: **Editar -> Añadir -> Muestreador -> Petición HTTP.**



A screenshot of the 'Petición HTTP' (HTTP Request) configuration dialog. The 'Basic' tab is selected. The 'Nombre:' field contains 'Login Alumnos'. The 'Servidor Web' section shows 'Protocolo:' (Protocol), 'Nombre de Servidor o IP:' (Server Name or IP), and 'Puerto:' (Port). The 'Petición HTTP' section shows 'Método:' (Method) set to 'POST', 'Ruta:' (Path) set to '/api/v1/auth/login', and 'Codificación del contenido:' (Content Encoding). Under 'Parameters', there are two entries: 'login' with value '\${login}' and 'password' with value '\${password}'. Both have 'Codificar?' (Encode?) checked and 'Content-Type' set to 'text/plain'. The 'Advanced' tab is also visible at the bottom.

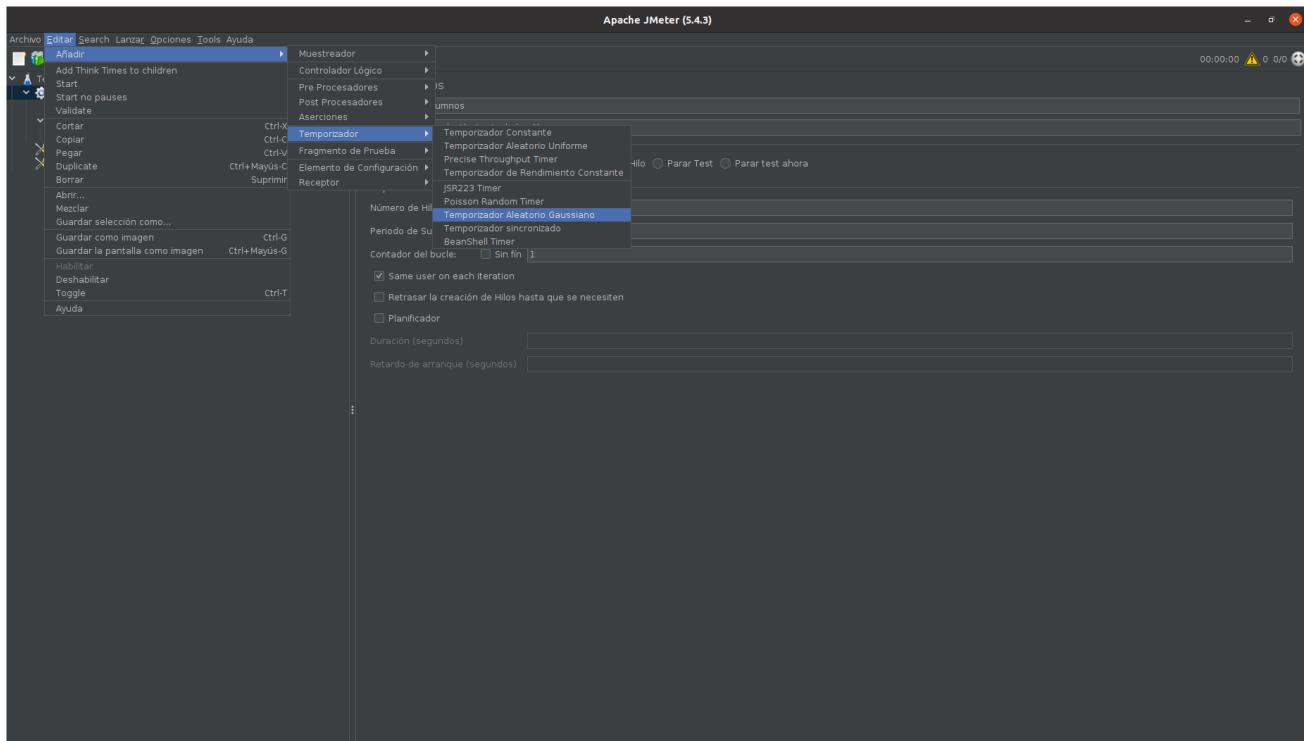
- 7) Cuando hacemos una autenticación válida, el servidor nos devuelve un mensaje OK además de un token. Vamos a **recuperar dicho token**.

Nos situamos en **Login Alumnos** y luego: **Editar -> Añadir -> Post Procesadores -> Extractor de Expresiones Regulares.**



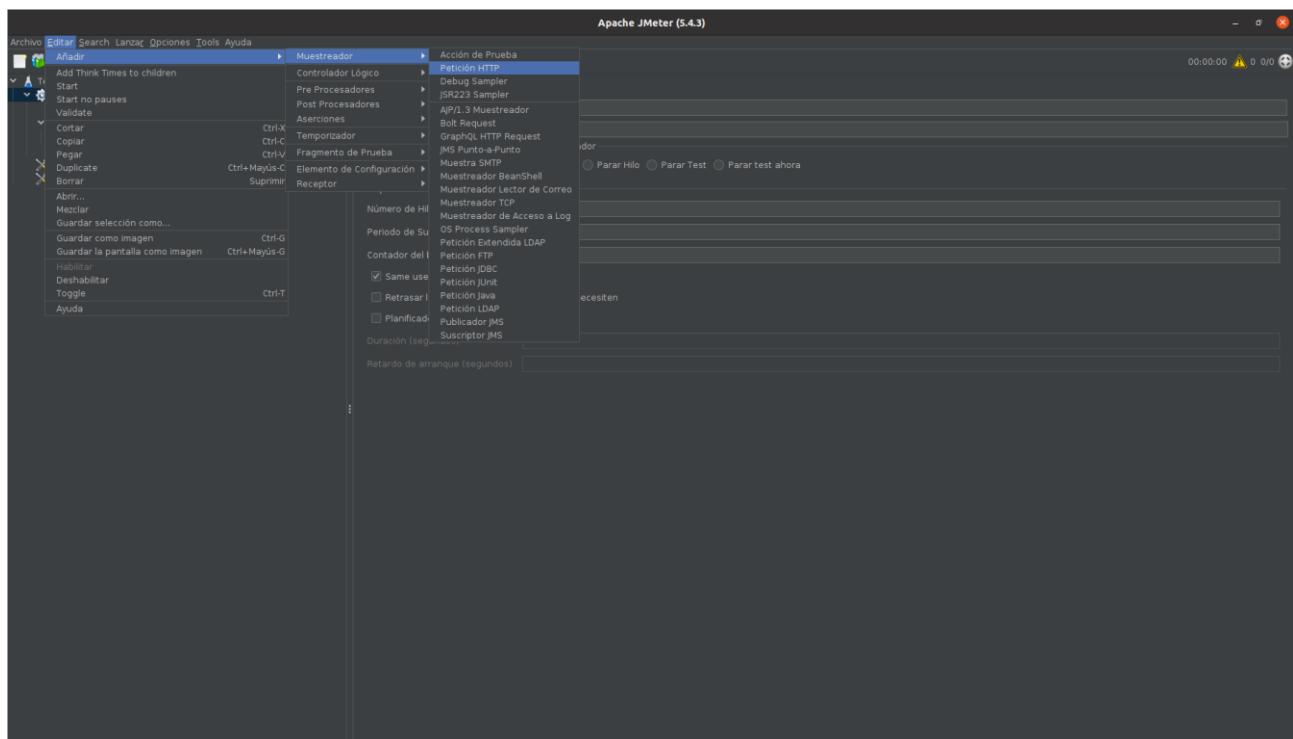
- 8) Para simular los accesos a la aplicación de una manera más realista, añadimos un temporizador que siga una distribución Gaussiana de números aleatorios.

Nos situamos en **Alumnos** y luego: **Editar -> Añadir -> Temporizador -> Temporizador Aleatorio Gaussiano.**



## 9) Recuperamos los datos de los alumnos creando una petición GET.

Nos situamos en **Alumnos** y luego: **Editar -> Añadir -> Muestreador -> Petición HTTP.**

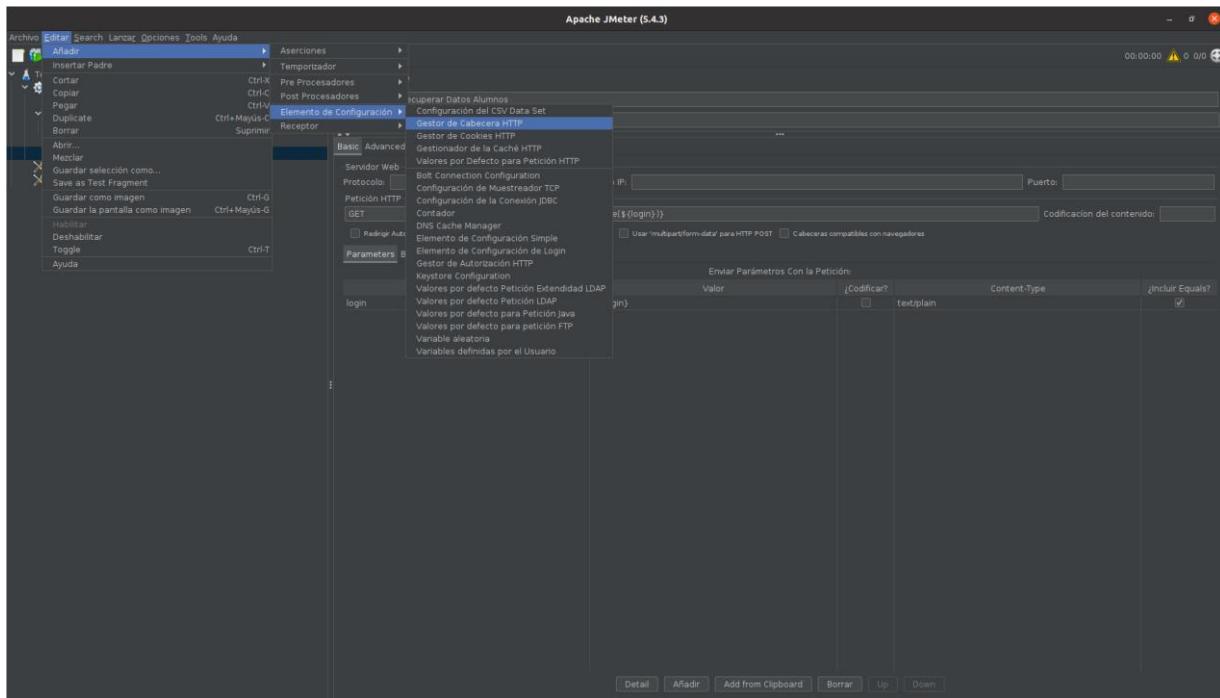
A detailed screenshot of the 'Recuperar Datos Alumnos' request configuration in JMeter. The 'Peticion HTTP' tab is active, showing the following details:

- Nombre:** Recuperar Datos Alumnos
- Protocolo:** Nombre de Servidor o IP: [empty], Puerto: [empty]
- Peticion HTTP:** Method: GET, Ruta: /api/v1/alumnos/alumno/\${\_\_urlencode(\${login})}, Codificación del contenido: [empty]
- Parameters:** login (Name: \${login}, Value: \${login}, Codificar?: checked, Content-Type: text/plain, Incluir Equals?: checked)

The left sidebar shows the test plan structure with 'Recuperar Datos Alumnos' highlighted.

**10) Resolver cualquier problema que haga que el token deje de ser válido usando el gestor de cabecera HTTP.**

Nos situamos en **Alumnos** y luego: **Editar -> Añadir -> Elemento de Configuración -> Gestor de Cabecera HTTP.**



Nombre:	Valor
Authorization	Bearer \$ (token)

**11)** Una vez terminada la configuración del grupo Alumnos, pasamos al grupo Administradores. **Creamos el grupo de hebras para los administradores** (en mi caso, le he asignado 2 hebras a dicho grupo).

Para crear un grupo de hebras, nos situamos en **Plan de Pruebas** y luego: **Editar -> Añadir -> Hilos (Usuarios) -> Grupo de Hilos.**

**12) Definir las credenciales para los Administradores.** Le indicamos que dichas credenciales se encuentran en el fichero **administradores.csv**, cuyos campos son usuario, contraseña.

Nos situamos en **Administradores** y luego: **Editar -> Añadir -> Elemento de Configuración -> Configuración del CSV Data Set.**

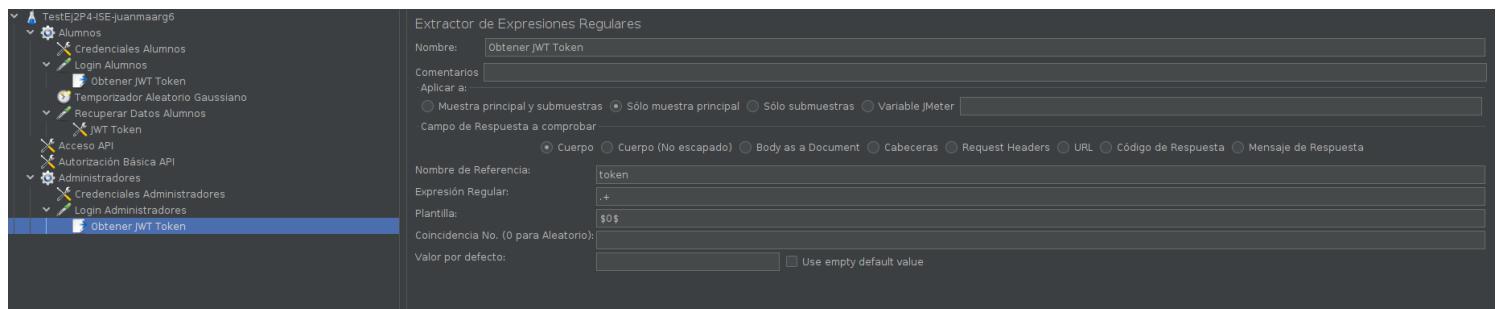
**13) Crear una petición POST para los Administradores.**

Nos situamos en **Administradores** y luego: **Editar -> Añadir -> Muestreador -> Petición HTTP.**

Enviar Parámetros Con la Petición:				
Nombre:	Valor	¿Codificar?	Content-Type	¿Incluir 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"/>

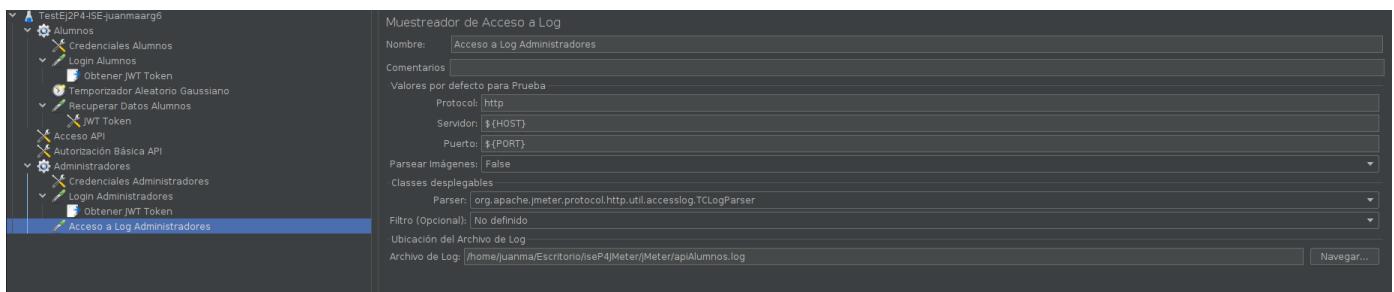
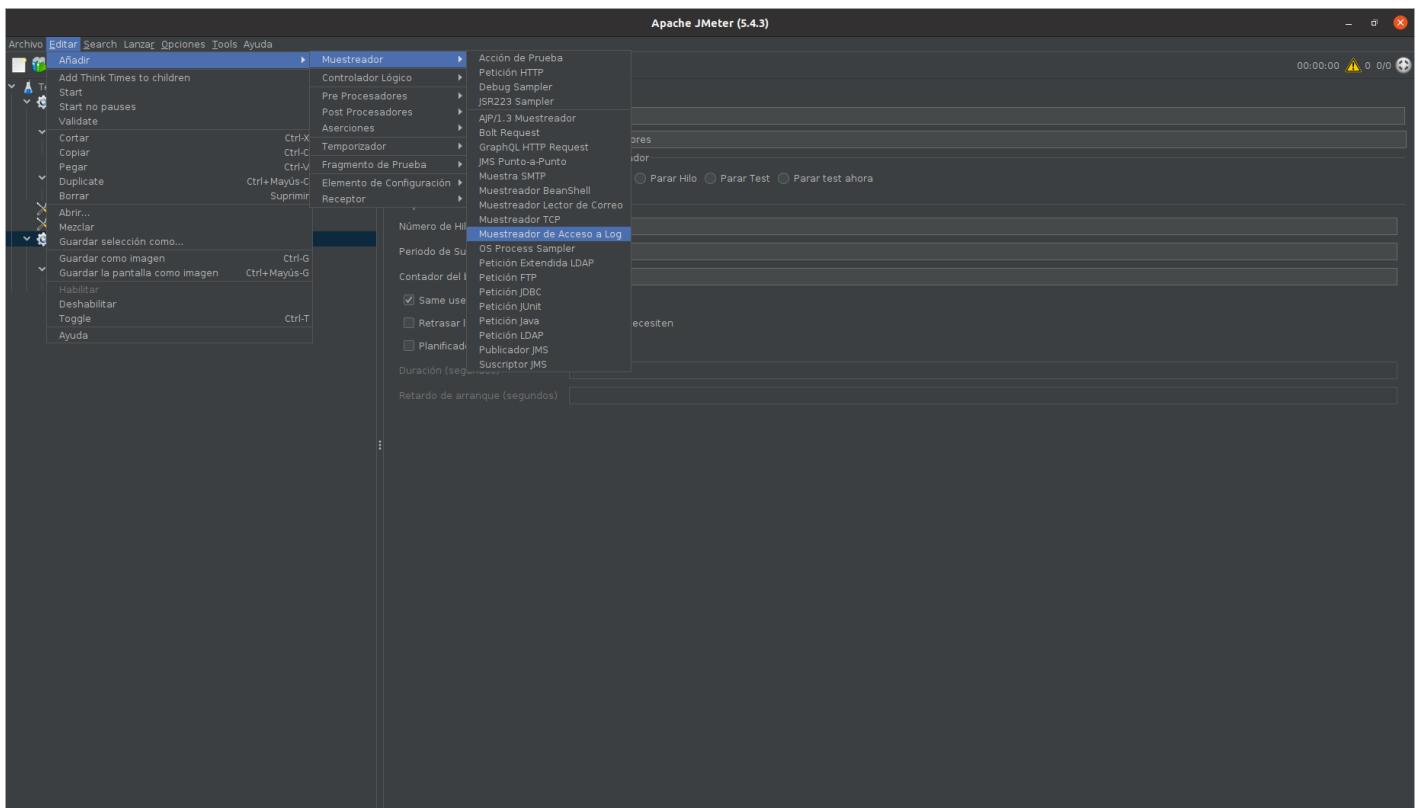
#### 14) Recuperar el token de los Administradores.

Nos situamos en **Login Administradores** y luego: **Editar -> Añadir -> Post Procesadores -> Extractor de Expresiones Regulares.**



#### 15) Como los Administradores pueden ver quién entra al sistema, es necesario añadir un ítem de muestreo de acceso a log.

Nos situamos en **Administradores** y luego: **Editar -> Añadir -> Muestreador -> Muestreador de Acceso a Log.**



- 16) Resolver cualquier problema que haga que el token deje de ser válido usando el gestor de cabecera HTTP.**

Nos situamos en **Acceso a Log Administradores** y luego: **Editar -> Añadir -> Elemento de Configuración -> Gestor de Cabecera HTTP.**



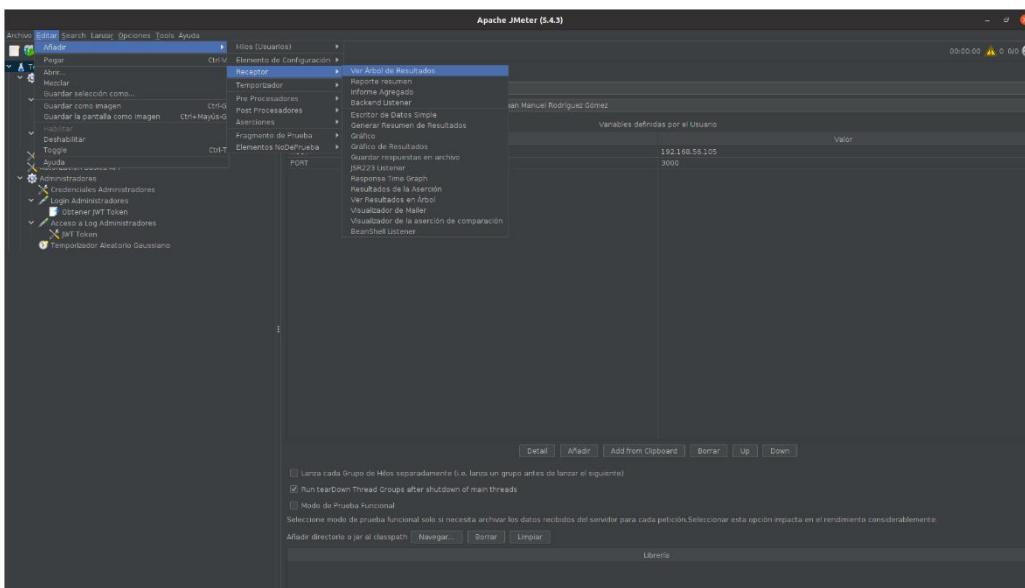
- 17) Añadimos un temporizador que siga una distribución Gaussiana de números aleatorios.**

Nos situamos en **Administradores** y luego: **Editar -> Añadir -> Temporizador -> Temporizador Aleatorio Gaussiano.**

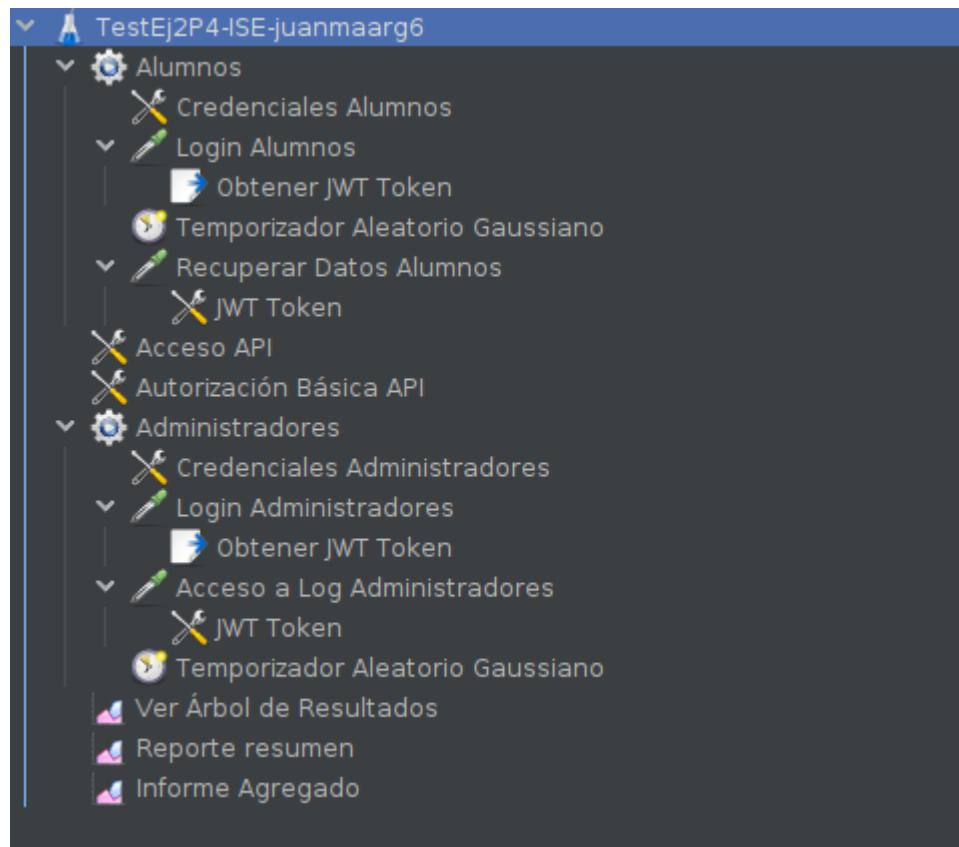


- 18) Añadimos ítems de generación de informe.**

Nos situamos en **Administradores** y luego: **Editar -> Añadir -> Receptor -> Ver Árbol de Resultados – Reporte Resumen – Informe Agregado.**



Una vez creado y configurado todo el test, el árbol que nos muestra los diferentes elementos que componen el test debe ser como se muestra a continuación:



## 2.5. Quinto Paso: Ejecución del Test en JMeter para la Aplicación iseP4JMeter

Levantamos la aplicación en la máquina de Ubuntu Server usando el comando `docker-compose up`.

```
[{"ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.18.0.3:47244","uuid":"4d9da44c-c10f-4433-a065-69f2dc99ed1","connectionId":11,"connectionCount":2}} mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:24.508+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":51800,"ctx":"conn11","msg":"client metadata","attr":{"remote":"172.18.0.3:47244","client":"conn11","doc":{"driver":{"name":"nodejs|Mongoose","version":"4.1.1"},"os":{"type":"Linux","name":"linux","architecture":"x64","version":"5.4.0-110-generic"},"platform":"Node.js v16.18.0, LE (unified)","version":"4.1.1|6.0.8"}}} mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:24.516+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":22943,"ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.18.0.3:47246","uuid":"16c70f51-87bb-4273-8472-776643cc03f67","connectionId":12,"connectionCount":3}} mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:24.520+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":51800,"ctx":"conn12","msg":"client metadata","attr":{"remote":"172.18.0.3:47246","client":"conn12","doc":{"driver":{"name":"nodejs|Mongoose","version":"4.1.1"},"os":{"type":"Linux","name":"linux","architecture":"x64","version":"5.4.0-110-generic"},"platform":"Node.js v16.18.0, LE (unified)","version":"4.1.1|6.0.8"}}} mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:24.518+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":22943,"ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.18.0.3:47276","uuid":"3921a36d-f580-4238-95a5-57c8ce56beaa","connectionId":13,"connectionCount":4}} mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:35.031+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":51800,"ctx":"conn13","msg":"client metadata","attr":{"remote":"172.18.0.3:47276","client":"conn13","doc":{"driver":{"name":"nodejs|Mongoose","version":"4.1.1"},"os":{"type":"Linux","name":"linux","architecture":"x64","version":"5.4.0-110-generic"},"platform":"Node.js v16.18.0, LE (unified)","version":"4.1.1|6.0.8"}}} mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:53.700+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":22430,"ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[165360743:700251] (1:0x7fa5c31b700), WT_SESSION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 1031, snapshot max: 1031 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 191'}}] mongod_1 | [{"t":1,"$date":"2022-05-26T23:23:55.514+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":5479200,"ctx":"TTLMonitor","msg":"Deleted expired documents using index","attr":{"namespace":"config.system.sessions","index":"lsidTLIndex","numDeleted":2,"durationMillis":118}}] mongod_1 | [{"t":1,"$date":"2022-05-26T23:24:54.753+00:00"}, {"s":1,"c":1,"e":1,"l":1,"id":22430,"ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[165360749:753072] (1:0x7fa5c31b700), WT_SESSION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 1036, snapshot max: 1036 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0) base write gen: 191'}}]
```

Luego, nos vamos a JMeter y para ejecutar el test le damos al botón señalado en la imagen inferior.



Hecho esto, podemos ver las peticiones realizadas a la aplicación en nuestra máquina de Ubuntu Server.

```
snapshot max: 1031 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0)
base write gen: 191'}
mongodb_1 | {"t":{"$date":"2022-05-26T23:23:55.514+00:00"},"s":"I", "c":"INDEX", "id":54792
00, "ctx":"TTLMonitor","msg":"Deleted expired documents using index","attr":{"namespace":"config.sys
tem.sessions","index":"lsidTTLIndex","numDeleted":2,"durationMillis":118}}
mongodb_1 | {"t":{"$date":"2022-05-26T23:24:54.753+00:00"},"s":"I", "c":"STORAGE", "id":22430
, "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1653607494:753072] [1:0x7fa55
c31b700], WT_SESSION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 1036,
snapshot max: 1036 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0)
base write gen: 191"}}
mongodb_1 | {"t":{"$date":"2022-05-26T23:25:56.200+00:00"},"s":"I", "c":"STORAGE", "id":22430
, "ctx":"Checkpointer","msg":"WiredTiger message","attr":{"message":"[1653607556:200963] [1:0x7fa55
c31b700], WT_SESSION.checkpoint: [WT_VERB_CHECKPOINT_PROGRESS] saving checkpoint snapshot min: 1039,
snapshot max: 1039 snapshot count: 0, oldest timestamp: (0, 0) , meta checkpoint timestamp: (0, 0)
base write gen: 191"}}
nodejs_1 | POST /api/v1/auth/login 200 346.259 ms - 196
nodejs_1 | POST /api/v1/auth/login 200 1169.682 ms - 183
nodejs_1 | POST /api/v1/auth/login 200 18.220 ms - 183
nodejs_1 | POST /api/v1/auth/login 200 11.660 ms - 183
nodejs_1 | POST /api/v1/auth/login 200 3.769 ms - 196
nodejs_1 | GET /api/v1/alumnos/alumno/daledalton%40tropoli.com?login=daledalton@tropoli.com 20
0 426.449 ms - 1172
nodejs_1 | GET /api/v1/alumnos/alumno/velezguthrie%40tropoli.com?login=velezguthrie@tropoli.co
m 200 33.370 ms - 1641
nodejs_1 | GET /api/v1/alumnos/alumno/rowlandsauunders%40tropoli.com?login=rowlandsauunders@trop
oli.com 200 37.934 ms - 740
mongodb_1 | {"t":{"$date":"2022-05-26T23:26:05.000+00:00"},"s":"I", "c":"NETWORK", "id":22943
, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.18.0.3:47544","uuid":"c95aafe
3-153b-418b-8519-d4ca3584aeac","connectionId":14,"connectionCount":53}}
mongodb_1 | {"t":{"$date":"2022-05-26T23:26:05.012+00:00"},"s":"I", "c":"NETWORK", "id":51800
, "ctx":"conn14","msg":"client metadata","attr":{"remote":"172.18.0.3:47544","client":"conn14","do
c":{"driver":{"name":"nodejs|Mongoose","version":"4.1.1"}, "os":{"type":"Linux","name":"linux","archi
tecture":"x64","version":"5.4.0-110-generic"}, "platform":"Node.js v16.13.0, LE (unified)"}, "version":
"4.1.1|6.0.8"}}
nodejs_1 | GET /api/v1/alumnos/alumno/deborawalker%40tropoli.com 200 37.031 ms - 805
nodejs_1 | GET /api/v1/alumnos/alumno/deborawalker%40tropoli.com 200 41.381 ms - 805
-
```

En los ítems de generación de informe, podemos ver que el test se ha ejecutado correctamente. Tanto en Reporte Resumen, como en Informe Agregado, se nos muestran todos los accesos realizados a la aplicación mediante el test, así como el porcentaje de error de autenticación, el cual se puede ver que es del 0 %.

The screenshot shows the JMeter interface with the 'TestEj2P4-ISE-juanmaarg6' test plan open. On the left, the tree view lists various requests under 'Alumnos' and 'Administradores' categories. On the right, two reports are displayed: 'Reporte resumen' and 'Informe Agregado'. Both reports have their names filled in the 'Nombre:' field. The 'Reporte resumen' report includes sections for 'Comentarios', 'Escribir todos los datos a Archivo', and 'Nombre de archivo'. The 'Informe Agregado' report also includes similar sections. Below the reports is a detailed table of performance metrics for each request, such as 'Media', 'Min', 'Max', 'Desv. Estándar', '% Error', 'Rendimiento', 'Kb/sec', 'Sent KB/sec', and 'Media de Bytes'.

This screenshot is identical to the one above, showing the 'Informe Agregado' report selected in the JMeter interface. It displays the same structure and data as the 'Reporte resumen' report, including the detailed performance table.

Por otro lado, en Ver Árbol de Resultados, se nos muestran todas las peticiones realizadas a la aplicación, así como las respuestas obtenidas. Por ejemplo, veamos el resultado obtenido por una petición GET para obtener los datos del alumno Rowland Gilliam Saunders.

The screenshot shows the 'Ver Árbol de Resultados' report selected in the JMeter interface. The left pane shows the test plan structure. The right pane displays the results of the 'Login Administradores' request, including the 'Texto' (Text) tab which shows the raw response data, and the 'Resultado del Muestreador' (Sampler Result) tab which provides detailed metrics like 'Nombre del hilo', 'Tiempo de carga', 'Latencia', and 'Tamaño en bytes'.

Tree View:

- Alumnos
  - Credenciales Alumnos
    - Login Alumnos
    - Obtener JWT Token
    - Temporizador Aleatorio Gaussiano
  - Recuperar Datos Alumnos
    - JWT Token
- Acceso API
- Autorización Básica API
- Administradores
  - Credenciales Administradores
    - Login Administradores
    - Obtener JWT Token
  - Acceso a Log Administradores
    - JWT Token
  - Temporizador Aleatorio Gaussiano
- Ver Árbol de Resultados
- Reporte resumen
- Informe Agregado

Resultados de la ejecución:

Nombre: Ver Árbol de Resultados

Comentarios: Escribir todos los datos a Archivo

Nombre de archivo:

Texto: Resultado del Muestreador Peticion Datos de Respuesta

Request Body: Cabeceras de petición:

```

1 GET http://192.168.56.105:3000/api/v1/alumnos/rowlandsaunders%40tropoli.com?login=rowlandsaunders@tropoli.com
2
3 GET data:
4
5
6 [no cookies]
7

```

Response Body: Cabeceras de respuesta:

Tree View:

- Alumnos
  - Credenciales Alumnos
    - Login Alumnos
    - Obtener JWT Token
    - Temporizador Aleatorio Gaussiano
  - Recuperar Datos Alumnos
    - JWT Token
- Acceso API
- Autorización Básica API
- Administradores
  - Credenciales Administradores
    - Login Administradores
    - Obtener JWT Token
  - Acceso a Log Administradores
    - JWT Token
  - Temporizador Aleatorio Gaussiano
- Ver Árbol de Resultados
- Reporte resumen
- Informe Agregado

Resultados de la ejecución:

Nombre: Ver Árbol de Resultados

Comentarios: Escribir todos los datos a Archivo

Nombre de archivo:

Texto: Resultado del Muestreador Peticion Datos de Respuesta

Request Body: Cabeceras de petición:

```

1 GET http://192.168.56.105:3000/api/v1/alumnos/rowlandsaunders%40tropoli.com?login=rowlandsaunders@tropoli.com
2
3 GET data:
4
5
6 [no cookies]
7

```

Response Body: Cabeceras de respuesta:

```

{
  "id": "628b5217fdce67eb97d23c3b7",
  "nombre": "Rowland",
  "apellidos": "Gilliam Saunders",
  "sexo": "male",
  "email": "rowlandsaunders@tropoli.com",
  "fechaNacimiento": "1984-09-27T00:00:00.000Z",
  "comentarios": "Consectetur incididunt nulla exercitation Lorem cillum est quis commodo commodo aute tempor occaecat. Veniam ipsa eiusmod veniam elit ex ipsum. Nostrud Lorem labore labore adipisciing ad. Occaecat reprehenderit amet reprehenderit pariatur molit pariatur. Paritatur Lorem magna dolor adipisciing voluptate labore exercitation ipsum nostrud irure.\n\n",
  "cursos": [
    {
      "id": "62900c8ca0ad5ea5ddafabc",
      "curso": "1",
      "media": 8.4
    },
    {
      "id": "62900c8ca0ad5ea5ddafabc",
      "curso": "2",
      "media": 6.4
    },
    {
      "id": "62900c8ca0ad5ea5ddafabc",
      "curso": "3",
      "media": 9
    }
  ],
  "usuario": "3"
}

```

### **3. Ejercicio Opcional**

#### **3.0. Enunciado del Ejercicio**

*Tras ver la parte de la práctica relacionada con contenedores, pruebe a ejecutar uno de los benchmarks de los seleccionados arriba y analice las diferencias en los resultados obtenidos en la MV directamente. Puede lanzar un contenedor con la imagen de phoronix, consulte: <https://www.phoronix.com/scan.php?page=article&item=docker-phoronix-pts&num=1> en su anfitrión (en la máquina virtual se quedará sin espacio). Otra posibilidad es ejecutar una imagen de contenedor de Ubuntu y ahí instalar phoronix.*

#### **3.1. Primer Paso: Instalación de Docker en nuestra Máquina Anfitrión**

Vamos a instalar Docker en nuestra máquina anfitrión. Para ello, al igual que hicimos para la instalación en Ubuntu Server, seguimos los pasos indicados en el guión de la práctica.

Añadimos la clave GPG para validar el repositorio de Docker:

```
juanma@Lenovo-JuanmaLinux:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
OK  
juanma@Lenovo-JuanmaLinux:~$ █
```

Añadimos el repositorio:

```
juanma@Lenovo-JuanmaLinux:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"  
Des:1 https://download.docker.com/linux/ubuntu focal InRelease [57,7 kB]  
Obj:2 http://es.archive.ubuntu.com/ubuntu focal InRelease  
Obj:3 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease  
Obj:4 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease  
Obj:5 http://security.ubuntu.com/ubuntu focal-security InRelease  
Des:6 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [16,7 kB]  
Descargados 74,3 kB en 1s (93,8 kB/s)  
Leyendo lista de paquetes... Hecho  
juanma@Lenovo-JuanmaLinux:~$ █
```

Ahora actualizamos los repositorios de Ubuntu Server:

```
juanma@Lenovo-JuanmaLinux:~$ sudo apt update  
Obj:1 http://es.archive.ubuntu.com/ubuntu focal InRelease  
Obj:2 https://download.docker.com/linux/ubuntu focal InRelease  
Obj:3 http://es.archive.ubuntu.com/ubuntu focal-updates InRelease  
Obj:4 http://es.archive.ubuntu.com/ubuntu focal-backports InRelease  
Obj:5 http://security.ubuntu.com/ubuntu focal-security InRelease  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
Se pueden actualizar 60 paquetes. Ejecute «apt list --upgradable» para verlos.  
juanma@Lenovo-JuanmaLinux:~$ █
```

Buscamos en el gestor el paquete docker-ce (Community Edition) y lo instalamos:

```
juanma@Lenovo-JuanmaLinux:~$ apt search docker-ce
Ordenando... Hecho
Buscar en todo el texto... Hecho
docker-ce/focal 5:20.10.16~3-0~ubuntu-focal amd64
  Docker: the open-source application container engine

docker-ce-cli/focal 5:20.10.16~3-0~ubuntu-focal amd64
  Docker CLI: the open-source application container engine

docker-ce-rootless-extras/focal 5:20.10.16~3-0~ubuntu-focal amd64
  Rootless support for Docker.
```

```
juanma@Lenovo-JuanmaLinux:~$ sudo apt install docker-ce
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  libssl-ttf2.0-0
Utilice «sudo apt autoremove» para eliminarlo.
Se instalarán los siguientes paquetes adicionales:
  containerd.io docker-ce-cli docker-ce-rootless-extras docker-scan-plugin pigz slirp4netns
Paquetes sugeridos:
  aufs-tools cgroups-mount | cgroup-lite
Se instalarán los siguientes paquetes NUEVOS:
  containerd.io docker-ce docker-ce-cli docker-ce-rootless-extras docker-scan-plugin pigz slirp4netns
0 actualizados, 7 nuevos se instalarán, 0 para eliminar y 60 no actualizados.
Se necesita descargar 102 MB de archivos.
Se utilizarán 422 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] S
Des:1 http://es.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57,4 kB]
Des:2 https://download.docker.com/linux/ubuntu focal/stable amd64 containerd.io amd64 1.6.4-1 [28,1 MB]
Des:3 https://es.archive.ubuntu.com/ubuntu focal/universe amd64 slirp4netns amd64 0.4.3-1 [74,3 kB]
Des:4 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce-cli amd64 5:20.10.16~3-0~ubuntu-focal [40,6 MB]
Des:5 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce amd64 5:20.10.16~3-0~ubuntu-focal [21,0 MB]
Des:6 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-ce-rootless-extras amd64 5:20.10.16~3-0~ubuntu-focal [8.168 kB]
Des:7 https://download.docker.com/linux/ubuntu focal/stable amd64 docker-scan-plugin amd64 0.17.0~ubuntu-focal [3.521 kB]
Descargados 102 MB en 6s (17,2 MB/s)
Seleccionando el paquete pigz previamente no seleccionado.
(Leyendo la base de datos ... 306042 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../0-pigz_2.4-1_amd64.deb ...
Desempaquetando pigz (2.4-1) ...
Seleccionando el paquete containerd.io previamente no seleccionado.
Preparando para desempaquetar .../1-containerd.io_1.6.4-1_amd64.deb ...
Desempaquetando containerd.io (1.6.4-1) ...
Seleccionando el paquete docker-ce-cli previamente no seleccionado.
Preparando para desempaquetar .../2-docker-ce-cli_5%3a20.10.16~3-0~ubuntu-focal_amd64.deb ...
```

Comprobamos el estado del servicio Docker (en nuestro caso, ya está activo. En el caso de que esté inactivo, lo iniciamos con el comando *sudo systemctl start docker*).

```
juanma@Lenovo-JuanmaLinux:~$ systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2022-05-20 13:38:35 CEST; 35s ago
    TriggeredBy: ● docker.socket
      Docs: https://docs.docker.com
       Main PID: 6051 (dockerd)
          Tasks: 13
         Memory: 31.5M
        CGroup: /system.slice/docker.service
                  └─6051 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.419180352+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.419229283+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.419243275+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.419615628+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.604933569+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.671868114+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.689884943+02:00" le
may 20 13:38:35 Lenovo-JuanmaLinux systemd[1]: Started Docker Application Container Engine.
may 20 13:38:35 Lenovo-JuanmaLinux dockerd[6051]: time="2022-05-20T13:38:35.739209257+02:00" le
lines 1-21/21 (END)
```

Ahora añadimos nuestro usuario al grupo docker:

```
juanma@Lenovo-JuanmaLinux:~$ sudo usermod -aG docker juanma
juanma@Lenovo-JuanmaLinux:~$ groups juanma
juanma : juanma adm cdrom sudo dip plugdev lpadmin lxd sambashare docker
juanma@Lenovo-JuanmaLinux:~$
```

Comprobamos la información de Docker y ejecutamos un script de prueba:

```
juanma@Lenovo-JuanmaLinux:~$ docker info ; docker run hello-world
Client:
  Context:    default
  Debug Mode: false
  Plugins:
    app: Docker App (Docker Inc., v0.9.1-beta3)
    buildx: Docker Buildx (Docker Inc., v0.8.2-docker)
    scan: Docker Scan (Docker Inc., v0.17.0)

Server:
ERROR: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: permission denied
errors pretty printing info
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
juanma@Lenovo-JuanmaLinux:~$
```

Nos sale el mismo error que en Ubuntu Server. Como ya comentamos, para solucionarlo tenemos que reiniciar nuestra sesión de usuario con el comando *su – juanma* en mi caso o, alternativamente, reiniciar el sistema. Una vez hecho esto, al comprobar la información de Docker e intentar ejecutar el script de prueba, vemos que todo se realiza correctamente.

```

juanma@Lenovo-JuanmaLinux:~$ su - juanma
Contraseña:
juanma@Lenovo-JuanmaLinux:~$ docker info ; docker run hello-world
Client:
  Context:    default
  Debug Mode: false
  Plugins:
    app: Docker App (Docker Inc., v0.9.1-beta3)
    buildx: Docker Buildx (Docker Inc., v0.8.2-docker)
    scan: Docker Scan (Docker Inc., v0.17.0)

Server:
  Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
  Images: 0
  Server Version: 20.10.16
  Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
  Logging Driver: json-file
  Cgroup Driver: cgroups
  Cgroup Version: 1
  Plugins:
    Volume: local
    Network: bridge host ipvlan macvlan null overlay
    Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
  Swarm: inactive
  Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
  Default Runtime: runc
  Init Binary: docker-init
  containerd version: 212e8b6fa2f44b9c21b2798135fc6fb7c53efc16
  runc version: v1.1.1-0-g52de29d
  init version: de40ad0
  Security Options:
    apparmor
    seccomp
      Profile: default
  Kernel Version: 5.13.0-41-generic
  Operating System: Ubuntu 20.04.4 LTS
  OSType: linux
  Architecture: x86_64
  CPUs: 8
  Total Memory: 15.52GiB
  Name: Lenovo-JuanmaLinux
  ID: 2JXF:R6RH:WZXL:F5VP:GGJI:6ZDA:GSF2:JL4F:554Q:KJFI:VGA5:6HFM
  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

Runtimes: io.containerd.runc.v2 io.containerd.runtime.v1.linux runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 212e8b6fa2f44b9c21b2798135fc6fb7c53efc16
runc version: v1.1.1-0-g52de29d
init version: de40ad0
Security Options:
  apparmor
  seccomp
    Profile: default
  Kernel Version: 5.13.0-41-generic
  Operating System: Ubuntu 20.04.4 LTS
  OSType: linux
  Architecture: x86_64
  CPUs: 8
  Total Memory: 15.52GiB
  Name: Lenovo-JuanmaLinux
  ID: 2JXF:R6RH:WZXL:F5VP:GGJI:6ZDA:GSF2:JL4F:554Q:KJFI:VGA5:6HFM
  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

Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:80f31da1ac7b312ba29d65080fddff797dd76acf870e677f390d5acba9741b17
Status: Downloaded newer image for hello-world:latest

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/
juanma@Lenovo-JuanmaLinux:~$ 
```

Una vez instalado Docker en nuestra máquina anfitrión, vamos a lanzar un contenedor con la imagen de Phoronix. Para ello, consultamos la siguiente página web donde se nos explica todo:

<https://www.phoronix.com/scan.php?page=article&item=docker-phoronix-pts&num=1>

Al final de la página web, nos dice cuál es el comando que debemos ejecutar para lanzar un contenedor de Docker con la imagen de Phoronix. Dicho comando es el siguiente:

*docker run -it phoronix/pts*

```
juanma@Lenovo-JuanmaLinux:~$ docker run -it phoronix/pts
Unable to find image 'phoronix/pts:latest' locally
latest: Pulling from phoronix/pts
8e5c1b329fe3: Pull complete
385c9d7e212b: Pull complete
cdacea015097: Pull complete
a42d94ebd1fc: Pull complete
472630ee4d9f: Pull complete
Digest: sha256:cf4f4fc6c294670ec3fe7fc3f0583c59d5329b93693a6a2e9ba282f83c8bc7a
Status: Downloaded newer image for phoronix/pts:latest

Updated OpenBenchmarking.org Repository Index
pts: 488 Distinct Tests, 2019 Test Versions, 56 Suites
Available Changes From 13 February To 20 May
Updated Test: pts/aom-av1 v3.3.0 AOM AV1
Updated Test: pts/apache v2.0.1 Apache HTTP Server
Updated Test: pts/avifenc v1.2.0 libavif avifenc
Updated Test: pts/build-mplayer v1.5.0 Timed MPlayer Compilation
Updated Test: pts/build-wasmer v1.1.0 Timed Wasmer Compilation
Updated Test: pts/david v1.12.0 dav1d
Updated Test: pts/ethr v1.1.0 Ethr
New Test: pts/fast-cli v1.0.0 fast-cli
Updated Test: pts/gravitymark v1.5.0 GravityMark
Updated Test: pts/gromacs v1.7.0 GROMACS
Updated Test: pts/influxdb v1.0.1 InfluxDB
Updated Test: pts/java-jmh v1.0.1 Java JMH
Updated Test: pts/mentier-benchmark v1.1.0 Memtier_benchmark
Updated Test: pts/mysqlslap v1.3.0 MariaDB
Updated Test: pts/nginx v2.0.1 nginx
Updated Test: pts/onednn v1.8.0 oneDNN
Updated Test: pts/onnx v1.5.0 ONNX Runtime
Updated Test: pts/ospray v2.9.0 OSPRAY
New Test: pts/ospray-studio v1.0.1 OSPRAY Studio
Updated Test: pts/perf-bench v1.0.4 perf-bench
Updated Test: pts/qmcpack v1.5.0 QMCPACK
Updated Test: pts/renaissance v1.3.0 Renaissance
Updated Test: pts/rocksdb v1.2.0 Facebook RocksDB
New Test: pts/specviewperf2020 v1.0.0 SPECViewPerf 2020
New Test: pts/speedtest-cli v1.0.0 speedtest-cli
Updated Test: pts/stress-ng v1.5.1 Stress-NG
Updated Test: pts/svt-av1 v2.5.0 SVT-AV1
Updated Test: pts/tensorflow-lite v1.1.0 TensorFlow Lite
Updated Test: pts/webp2 v1.1.0 WebP Image Encode
New Suite: pts/internet-speed v1.0.0 Internet Speed
Updated Suite: pts/oneapi v1.3.3 Intel oneAPI
Updated Suite: pts/raytracing v1.0.2 Raytracing
Updated OpenBenchmarking.org Repository Index
system: 41 Distinct Tests, 122 Test Versions
Available Changes From 13 February To 20 May
Updated Test: system/blender v1.2.1 Blender
Updated Test: system/selenium v1.0.29 Selenium
New Test: system/timed-battery-test v1.0.0 Timed Battery Test
Updated OpenBenchmarking.org Repository Index
git: 8 Distinct Tests, 11 Test Versions
Available Changes From 13 February To 20 May
Updated Test: ait/david v1.1.0 dav1d
```

```

New Test: system/timed-battery-test v1.0.0 Timed Battery Test
Updated OpenBenchmarking.org Repository Index
git: 8 Distinct Tests, 11 Test Versions
Available Changes From 13 February To 20 May
Updated Test: git/david v1.1.0 david

Phoronix Test Suite v10.8.3
Interactive Shell

Generating Shell Cache...
Refreshing OpenBenchmarking.org Repository Cache...

PROCESSOR:
Core Count: 4
Thread Count: 8
Extensions: SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE
Cache Size: 6 MB
Microcode: 0xea
Core Family: Kaby/Coffee/Whiskey Lake
Scaling Driver: intel_pstate powersave (EPP: balance_performance)

GRAPHICS:
Frequency: 1100MHz
Screen: 1920x1080

MOTHERBOARD:
BIOS Version: 4KCN40WN
Audio: Realtek ALC233

MEMORY:
16GB

DISK:
File-System: overlayfs
Disk Scheduler: NONE

OPERATING SYSTEM:
Kernel: Ubuntu 20.04 LTS
5.13.0-41-generic (x86_64)
Docker
itlb_multithit: KVM: Mitigation of VMX disabled
+ l1tf: Mitigation of PTE Inversion; VMX: conditional cache flushes SMT vulnerable
+ mds: Mitigation of Clear buffers; SMT vulnerable
+ meltdown: Mitigation of PTI
+ spec_store_bypass: Mitigation of SSB disabled via prctl and seccomp
+ spectre_v1: Mitigation of usercopy/swaps barriers and _user pointer sanitization
+ spectre_v2: Mitigation of Retpolines IBRS_FW STIBP: conditional RSB filling
+ srbs: Mitigation of Microcode
+ tsx_async_abort: Not affected

CPU Temperature: 45.00 C CPU Usage (Summary): 8.38 % Memory Usage: 1853 MB System Temperature: 44.5 C System Uptime 8 M

Phoronix Test Suite command to run or help for all possible options, commands for a quick overview of options, interactive for a guided experience, system
n, exit to exit. For new users, benchmark is the simplest and most important sub-command. Tab auto-completion support available.

# phoronix-test-suite

```

## 3.2. Segundo Paso: Ejecución del Benchmark Sudokut en el Contenedor

### Docker con la Imagen de Phoronix

Como el enunciado solo nos indica ejecutar y comparar resultados con uno de los benchmarks elegidos en el primer ejercicio, vamos a ejecutar solo el benchmark Sudokut. Para ello, lanzamos el comando *benchmark sudokut* y esperamos un poco a que el propio programa instale las dependencias necesarias para ejecutarse.

```

# phoronix-test-suite benchmark sudokut
  Evaluating External Test Dependencies ......

The following dependencies are needed and will be installed:
  - tcl
  - tclsh

This process may take several minutes.
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot
debconf: falling back to frontend: Readline
Reading package lists...
Building dependency tree...
Reading state information...
The following additional packages will be installed:
  libtcl8.6 tcl8.6
Suggested packages:
  tcl-tclreadline
The following NEW packages will be installed:
  libtcl8.6 tcl tcl8.6
0 upgraded, 3 newly installed, 0 to remove and 1 not upgraded.
Need to get 922 kB of archives.
After this operation, 4197 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/main amd64 libtcl8.6 amd64 8.6.10+dfsg-1 [968 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal/main amd64 tcl8.6 amd64 8.6.10+dfsg-1 [14.8 kB]
Get:3 http://archive.ubuntu.com/ubuntu focal/universe amd64 tcl amd64 8.6.9+1 [5112 B]
Fetched 922 kB in 0s (635 kB/s)
Selecting previously unselected package libtcl8.6:amd64.
(Reading database ... 10667 files and directories currently installed.)
Preparing to unpack .../libtcl8.6_8.6.10+dfsg-1_amd64.deb ...
Unpacking libtcl8.6:amd64 (8.6.10+dfsg-1) ...
Selecting previously unselected package tcl8.6.
Preparing to unpack .../tcl8.6_8.6.10+dfsg-1_amd64.deb ...
Unpacking tcl8.6 (8.6.10+dfsg-1) ...
Selecting previously unselected package tcl.
Preparing to unpack .../archives/tcl_8.6.9+1_amd64.deb ...
Unpacking tcl (8.6.9+1) ...
Setting up libtcl8.6:amd64 (8.6.10+dfsg-1) ...
Setting up tcl8.6 (8.6.10+dfsg-1) ...
Setting up tcl (8.6.9+1) ...
Processing triggers for libc-bin (2.31-0ubuntu9.7) ...
To Install: pts/sudokut-1.0.1

```

Una vez terminada la instalación de las dependencias, nos preguntará si queremos guardar los resultados en un fichero. Diremos que sí.

```
pts/sudokut-1.0.1:  
  Test Installation 1 of 1  
  1 File Needed [0.02 MB]  
  Downloading: sudokut0.4-1.tar.bz2  
  Downloading .....  
  Approximate Install Size: 0.1 MB  
  Estimated Install Time: 2 Seconds  
  Installing Test @ 11:57:34  
  
System Information  
  
PROCESSOR: Intel Core i7-7700HQ @ 3.80GHz  
Core Count: 4  
Thread Count: 8  
Extensions: SSE 4.2 + AVX2 + AVX + RDRAND + FSGSBASE  
Cache Size: 6 MB  
Microcode: 0xea  
Core Family: Kaby/Coffee/Whiskey Lake  
Scaling Driver: intel_pstate powersave (EPP: balance_performance)  
  
GRAPHICS: i915drmfb  
Frequency: 1100MHz  
Screen: 1920x1080  
  
MOTHERBOARD: LENOVO LNVNB161216  
BIOS Version: 4KCN40WW  
Audio: Realtek ALC233  
  
MEMORY: 16GB  
  
DISK: 256GB SK hynix HFS256GD9TNG-62A0A + 1000GB TOSHIBA MQ04ABF1  
File-System: overlayfs  
Disk Scheduler: NONE  
  
OPERATING SYSTEM: Ubuntu 20.04.4 LTS  
Kernel: 5.13.0-41-generic (x86_64)  
System Layer: Docker  
Security:  
  + itlb_mitigations: KVM: Mitigation of VMX disabled  
  + l1tf: Mitigation of PTE Inversion; VMX: conditional cache flushes  
  + mds: Mitigation of Clear buffers; SMT vulnerable  
  + meltdown: Mitigation of PTI  
  + spec_store_bypass: Mitigation of SSB disabled via prctl and seccomp  
  + spectre_v1: Mitigation of usercopy/swaps barriers and __usercopy  
  + spectre_v2: Mitigation of Retpolines IBPB: conditional IBRS_FW  
  + srbds: Mitigation of Microcode  
  + tsx_async_abort: Not affected  
  
Would you like to save these test results (Y/n): █
```

```
Would you like to save these test results (Y/n): Y  
Enter a name for the result file: benchmark-sudokut-localhost  
Enter a unique name to describe this test run / configuration: benchmarkLocal  
  
If desired, enter a new description below to better describe this result set / system configuration.  
Press ENTER to proceed without changes.  
  
Current Description: Docker testing on Ubuntu 20.04.4 LTS via the Phoronix Test Suite.  
New Description: Ejercicio Opcional ISE
```

Tras ejecutarse el benchmark obtenemos el siguiente resultado:



Nos pide si queremos ver los resultados almacenados en el archivo que confirmamos antes que queríamos crear para almacenar los resultados:

```
Do you want to view the text results of the testing (Y/n): Y
benchmark-sudokut-localhost
Ejercicio Opcional ISE

benchmarkLocal:
Processor: Intel Core i7-7700HQ @ 3.80GHz (4 Cores / 8 Threads), Motherboard: LENOVO LNVNB161Z16 (4KCN40WW BIOS), Memory: 16GB, Disk: 256GB SK hynix HFS256GD9TNG-62A0A + 1000GB TOSHIBA MQ04ABF1,
Graphics: i915drmfb (1100MHz), Audio: Realtek ALC233
OS: Ubuntu 20.04.4 LTS, Kernel: 5.13.0-41-generic (x86_64), File-System: overlayfs, Screen Resolution: 1920x1080, System Layer: Docker

Sudokut 0.4
Total Time
Seconds < Lower Is Better
benchmarkLocal . 18.39 |=====
```

Finalmente, nos pide si queremos subir nuestros resultados de ejecución a OpenBenchmarking.org. Decimos que sí, y nos devuelve el enlace con los resultados de nuestra ejecución:

<https://openbenchmarking.org/result/2205205-NE-BENCHMARK63>

```
Would you like to upload the results to OpenBenchmarking.org (y/n): y
Would you like to attach the system logs (lspci, dmesg, lsusb, etc) to the test result (y/n): y

Results Uploaded To: https://openbenchmarking.org/result/2205205-NE-BENCHMARK63
```

# benchmark-sudokut-localhost

Ejercicio Opcional ISE

Compare your own system(s) to this result file with the [Phoronix Test Suite](#) by running the command: **phoronix-test-suite benchmark 2205205-NE-BENCHMARK63**

[Jump To Table - Results](#)

## Statistics

## Graph Settings Table

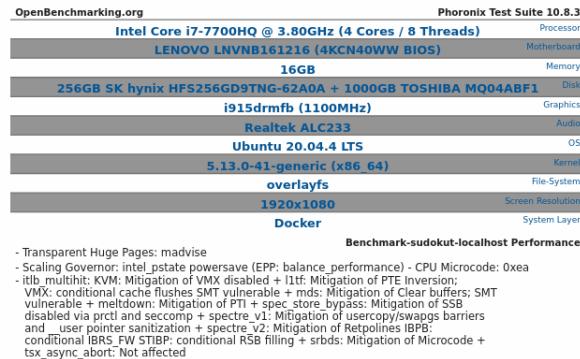
Remove Outliers Before Calculating Averages  Prefer Vertical Bar Graphs  Show Detailed System Result Table

## Run Management

RESULT	PERFORMANCE PER	DATE	TEST
IDENTIFIER	DOLLAR	RUN	DURATION
benchmarkLocal	0	May 20	1 Minute

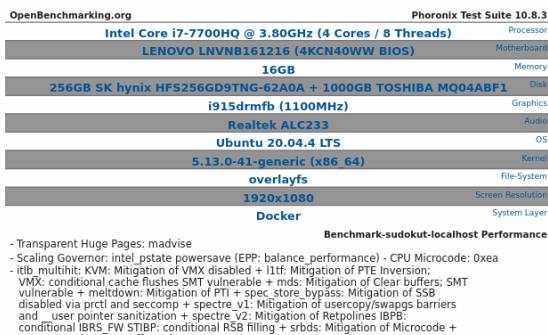
[Refresh Results](#)

## benchmark-sudokut-localhost



nix-test-suite

## benchmark-sudokut-localhost



System Logs

## Sudoku

This is a test of Sudokut, which is a Sudoku puzzle solver written in Tcl. This test measures how long it takes to solve 100 Sudoku puzzles. [Learn more via the OpenBenchmarking.org test page.](#)



En resumen, hemos obtenido los siguientes resultados tras ejecutar el benchmark Sudokut en nuestra máquina anfitrión en un contenedor de Docker con la imagen de Phoronix :

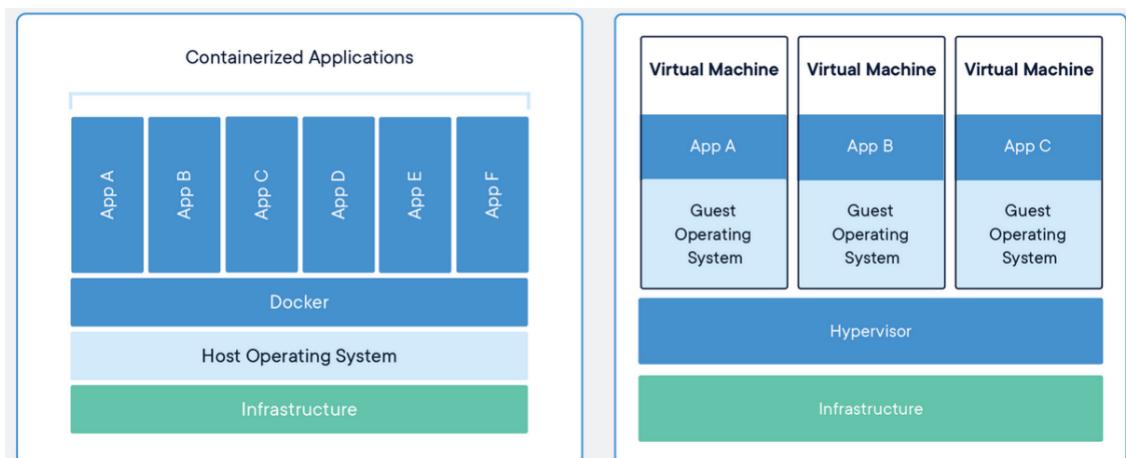
<b>Sistema Operativo</b>	<i>Ubuntu 20.04 (Contenedor Docker)</i>
<b>Benchmark</b>	<i>Sudokut</i>
<b>Tiempo medio resultante</b>	18.388 segundos
<b>Desviación típica</b>	2.06%

### **3.5. Tercer Paso: Comparativa de los Resultados del Benchmark Sudokut**

<b>Sistema Operativo</b>	<i>CentOS Linux 8</i>	<i>Ubuntu Server 20.04</i>	<i>Ubuntu 20.04 (Contenedor Docker)</i>
<b>Benchmark</b>	<i>Sudokut</i>	<i>Sudokut</i>	<i>Sudokut</i>
<b>Tiempo medio resultante</b>	13.07 segundos	12.85 segundos	18.388 segundos
<b>Desviación típica</b>	0.33%	3.30%	2.06%

Podemos observar que la ejecución del benchmark en el contenedor de Docker tiene un tiempo medio de ejecución considerablemente mayor al de las máquinas virtuales con CentOS y Ubuntu Server. Esto se debe a que una máquina virtual es una virtualización a nivel de hardware mientras que un contenedor es una virtualización a nivel de software, de manera que en los contenedores ahorraremos espacio, recursos, son más veloces en el arranque... pero el tiempo de cómputo es mayor. Para consultar más información acerca de las diferencias y ventajas entre máquinas virtuales y contenedores, se recomienda visitar el siguiente enlace:

<https://www.campusmvp.es/recursos/post/que-diferencia-hay-entre-docker-contenedores-y-maquinas-virtuales.aspx>



## **4. Referencias**

- <https://arstechnica.com/linux/performance-testing/phoronix-test-suite/>
- <https://stackoverflow.com/questions/60238534/add-libraries-for-static-compilation-centos-8>
- <https://openbenchmarking.org/tests>
- <https://openbenchmarking.org/test/pts/ramspeed>
- <https://openbenchmarking.org/test/pts/sudokut>
- <https://www.phoronix.com/forums/forum/phoronix/phoronix-test-suite/12701-quick-overall-system-performance-suite/page10>
- <https://www.drupaladicto.com/snippet/como-corregir-error-docker-got-permission-denied-while-trying-connect-docker-daemon-socket>
- <https://www.phoronix.com/scan.php?page=article&item=docker-phoronix-pts&num=1>
- <https://www.campusmvp.es/recursos/post/que-diferencia-hay-entre-docker-contenedores-y-maquinas-virtuales.aspx>