

---

## Práctica 4: Benchmarking y ajuste del sistema

---

Javier Gómez Luzón

# Índice

Phoronix.....	3
Ab.....	6
Jmeter.....	7

# Phoronix

En nuestro ubuntu server ejecutamos:

```
>sudo apt install phoronix-test-suite
```

```
>sudo apt install firefox
```

Luego yo instale zip y unzip porque si no los tenia, no podia descomprimir unos paquetes que se necesitaban.

```
>sudo apt-get install zip unzip
```

Es importante que comprobemos que la opción X11Forwarding en /etc/ssh/sshd\_config este habilitada.

Instalaremos un test y lo ejecutaremos:

```
>phoronix-test-suite install pts/sudokut
```

```
>phoronix-test-suite run pts/sudokut
```

Y este es el resultado que obtenemos:

```
Sudokut 0.4:
pts/sudokut-1.0.0
Test 1 of 1
Estimated Trial Run Count: 3
Estimated Time To Completion: 5 Minutes
Started Run 1 @ 00:38:57

Started Run 2 @ 00:39:21

Started Run 3 @ 00:39:42
[Std. Dev: 5.97%]
Started Run 4 @ 00:40:01
[Std. Dev: 5.07%]
Started Run 5 @ 00:40:20 [Std. Dev: 5.05%]
Started Run 6 @ 00:40:40 [Std. Dev: 7.95%]

Test Results:
17.410208940506
18.814695119858
16.749707937241
17.216130018234
18.572481870651
20.711945056915

Average: 18.25 Seconds

Would you like to upload the results to OpenBenchmarking.org (Y/n): Would you like to attach
the system logs (lspci, dmesg, lsusb, etc) to the test result (Y/n):
Results Uploaded To: https://openbenchmarking.org/result/1712185-KH-ATXT3640463
```

Instalaremos otro test y lo ejecutaremos:

```
>phoronix-test-suite install pts/apache
```

```
>phoronix-test-suite run pts/apache
```

Y este es el resultado que obtenemos:

```

Apache Benchmark 2.4.7:
  pts/apache-1.6.1
  Test 1 of 1
  Estimated Trial Run Count:    3
  Estimated Time To Completion: 5 Minutes
  Running Pre-Test Script @ 22:29:53
  Started Run 1 @ 22:29:58
  Started Run 2 @ 22:32:31
  Started Run 3 @ 22:35:03 [Std. Dev: 0.65%]
  Running Post-Test Script @ 22:37:33

  Test Results:
    6654.42
    6740.62
    6704.53

  Average: 6699.86 Requests Per Second

```

Ahora conectamos con nuestro ubuntu desktop a nuestro ubuntu server con ssh. La opción -X es para interfaz gráfica.

```
>ssh javi@192.168.58.105 -X
```

Y lanzamos la orden que ejecutara phoronix:

```
>phoronix-test-suite gui
```

```

Do you agree to these terms and wish to proceed (Y/n): Y
Enable anonymous usage / statistics reporting (Y/n): Y
Enable anonymous statistical reporting of installed software / hardware (Y/n): Y_

```

Nos saldrán las siguientes opciones, yo las he aceptado todas.

Se nos abrirá el navegador:



```

Connecting To WebSocket Server...
Starting Session...
Generating Phodevi Cache + VFS...
Starting Phodevi Sensor Handler...
Downloading Test Information...

```

Seleccionamos un test y lo ejecutamos

## Main Tests Results System

1 Test Queued To Benchmark



### BioShock Infinite

A Steam-based test of the BioShock Infinite game.

#### RUN THIS TEST

RESOLUTION	800 x 600 ▾
EFFECTS QUALITY	Low ▾
<b>ADD TEST TO RUN QUEUE</b>	

#### TEST PROFILE INFORMATION

TEST PROFILE	pts/bioshock-infinite-1.0.1
MAINTAINER	Michael Larabel
TEST TYPE	Graphics
SOFTWARE TYPE	Game
LICENSE TYPE	Free
TEST STATUS	Verified
PROJECT SITE	store.steampowered.com

#### INSTALLATION DATA

ESTIMATED TEST RUN-TIME	7 Minutes
LAST LOCAL RUN	19 December 2017
LATEST LOCAL RUN-TIME	N/A
TIMES RUN LOCALLY	1

Pulsamos en 'Add test to run queue'.

## Main Tests Results System

2 Tests Queued To Benchmark



Resolution: 800 x 600 - Effects Quality: Low

### BioShock Infinite

Resolution: 800 x 600 - Effects Quality: Low

Test Name: BIOSHOCK

Test Identifier: 1

Test Description:

With Phoronix Test Suite 5.0 your results will be automatically uploaded to [OpenBenchmarking.org](http://OpenBenchmarking.org) as we continue refining the features and new user-interface of PTS5 and complete a local results viewer interface. If you require behind-the-firewall or private testing support within the HTML5 UI, please contact us. The Phoronix Test Suite command-line interface will continue to operate as it always has been for either public or private testing.

**Start Benchmarking**

Pulsamos en 'Start Benchmarking'.

## Main Tests Results System

2 Tests Queued To Benchmark



### Benchmark Complete

The 1 run within **Bioshock** [bioshock] is complete.



# AB

En nuestro Ubuntu Desktop instalaremos ab con:

```
>sudo apt-get install apache2-utils
```

Lo ejecutaremos tanto para CentOS como para Ubuntu Server:

```
>ab -g datos_ubuntu.tsv -n 100 -c 100 http://192.168.58.105:80/index.html
```

```
Server Software:      Apache/2.4.18
Server Hostname:      192.168.58.120
Server Port:          80

Document Path:        /index.html
Document Length:      11321 bytes

Concurrency Level:    100
Time taken for tests:  0.111 seconds
Complete requests:    100
Failed requests:       0
Total transferred:    1159500 bytes
HTML transferred:     1132100 bytes
Requests per second:  899.01 [#/sec] (mean)
Time per request:      111.234 [ms] (mean)
Time per request:      1.112 [ms] (mean, across all concurrent requests)
Transfer rate:         10179.66 [Kbytes/sec] received

Connection Times (ms)
      min     mean[+/-sd] median   max
Connect:    2      29   8.2      34     38
Processing: 22      43  11.5     47     62
Waiting:    14      41  12.4     45     60
Total:      53      72   8.6     75     85

Percentage of the requests served within a certain time (ms)
 50%      75
 66%      77
 75%      79
 80%      81
 90%      82
 95%      84
 98%      85
 99%      85
100%     85 (longest request)
javi@javi-VirtualBox:~$
```

```
>ab -g datos_centos.tsv -n 100 -c 100 http://192.168.58.110:80/index.php
```

```
Server Software:      Apache/2.4.6
Server Hostname:      192.168.58.110
Server Port:          80

Document Path:        /index.php
Document Length:      23 bytes

Concurrency Level:    100
Time taken for tests:  0.105 seconds
Complete requests:    100
Failed requests:      0
Total transferred:    22600 bytes
HTML transferred:     2300 bytes
Requests per second:  955.05 [#/sec] (mean)
Time per request:     104.707 [ms] (mean)
Time per request:     1.047 [ms] (mean, across all concurrent requests)
Transfer rate:        210.78 [Kbytes/sec] received

Connection Times (ms)
      min      mean[+/-sd] median   max
Connect:    8       9   1.0      9    11
Processing: 13      52  22.1     52    88
Waiting:    13      51  22.2     52    87
Total:      24      61  21.2     61    95

Percentage of the requests served within a certain time (ms)
 50%    61
 66%    72
 75%    81
 80%    84
 90%    91
 95%    94
 98%    95
 99%    95
100%    95 (longest request)

javi@javi-VirtualBox:~$
```

## Jmeter

Jmeter necesita Java 8, que se instala así:

```
>sudo add-apt-repository ppa:webupd8team/java
```

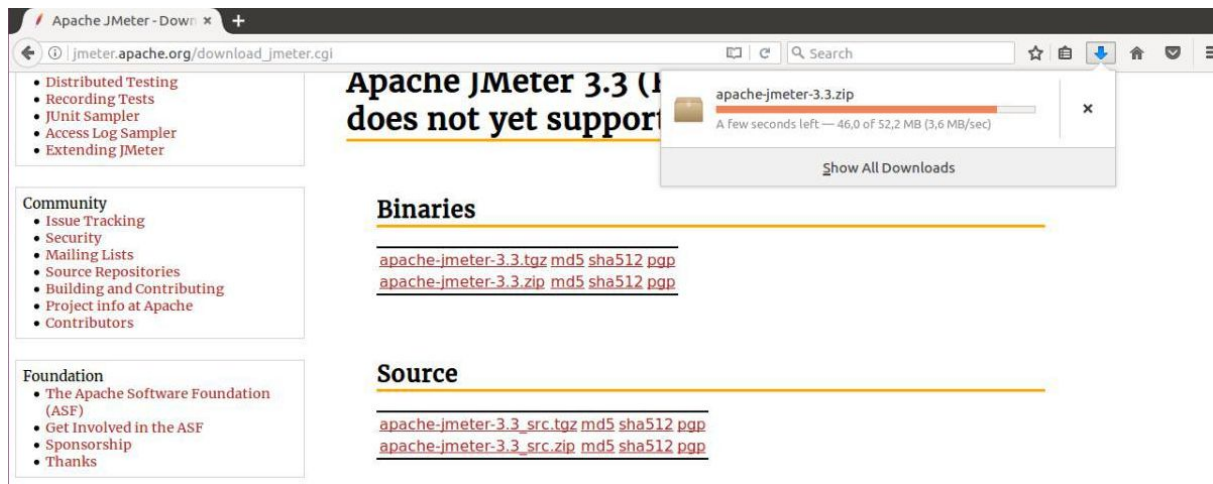
```
>sudo apt-get update
```

```
>sudo apt-get install oracle-java8-installer
```

Con esta última tendremos que simplemente aceptar las condiciones de java.

Ahora iremos a la dirección web: [http://jmeter.apache.org/download\\_jmeter.cgi](http://jmeter.apache.org/download_jmeter.cgi)





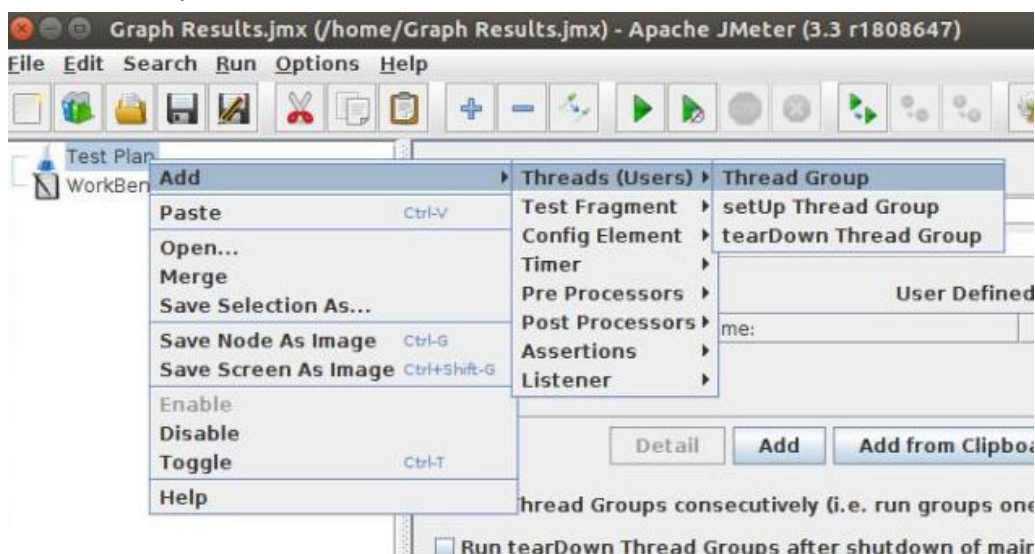
Como se puede ver, descargamos el paquete apache-jmeter-3.3.zip  
 Descomprimos el archivo y nos vamos al /bin de la carpeta descomprimida y  
 ejecutamos jmeter.

```
javi@javi-VirtualBox:~/Downloads/apache-jmeter-3.3/bin$ ./jmeter
=====
Don't use GUI mode for load testing, only for Test creation and Test debugging !
For load testing, use NON GUI Mode:
  jmeter -n -t [jmx file] -l [results file] -e -o [Path to output folder]
& adapt Java Heap to your test requirements:
  Modify HEAP="-Xms512m -Xmx512m" in the JMeter batch file
=====
Dec 17, 2017 7:19:18 PM java.util.prefs.FileSystemPreferences$1 run
INFO: Created user preferences directory.
```

Para empezar el Benchmarking seguiremos los pasos que se describen a  
 continuación:

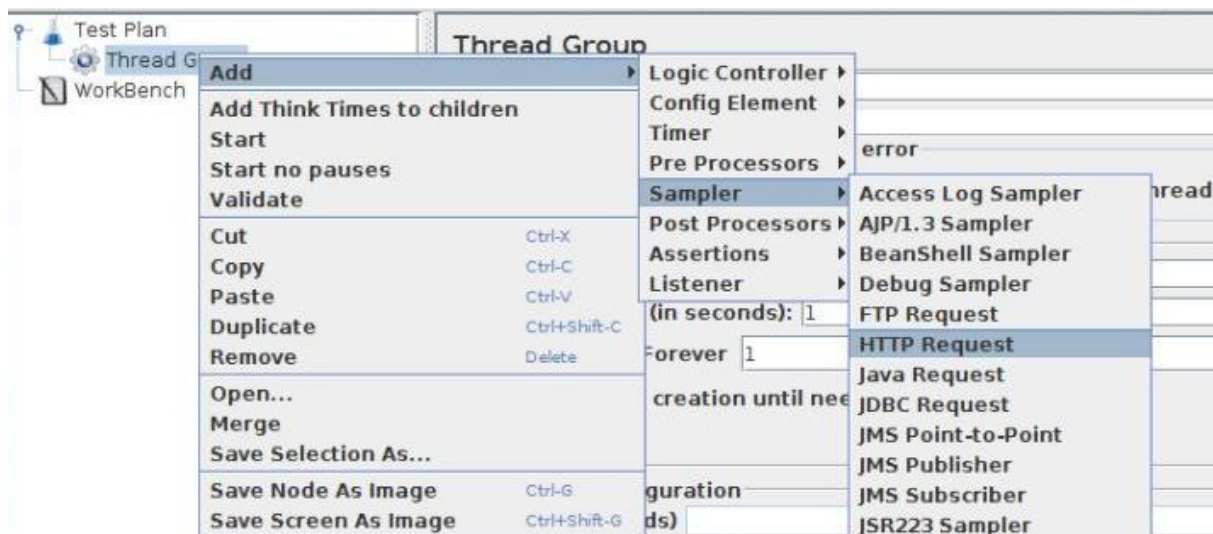
Test Plan->Add->Threads->Thread Group

Deberemos poner un numero alto de hilos en la hebra.

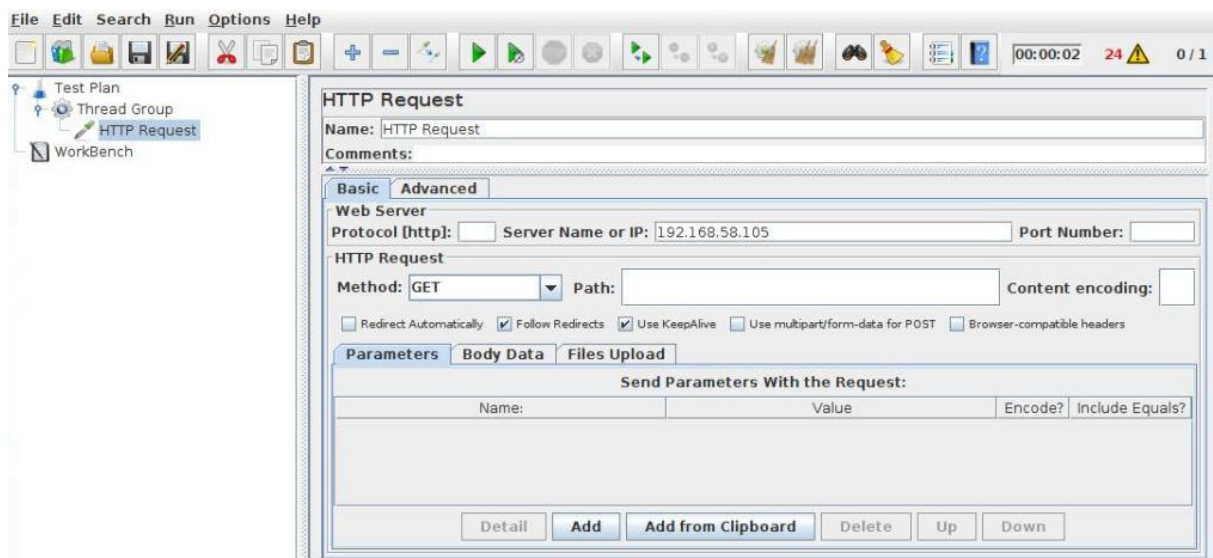




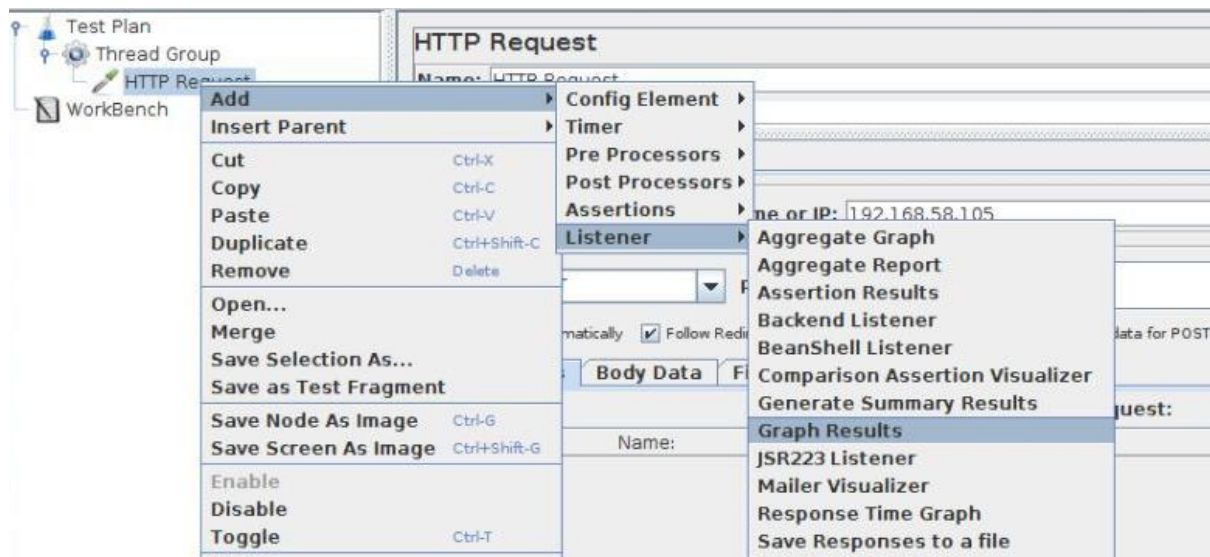
Thread Group->Add->Sampler->Http Request



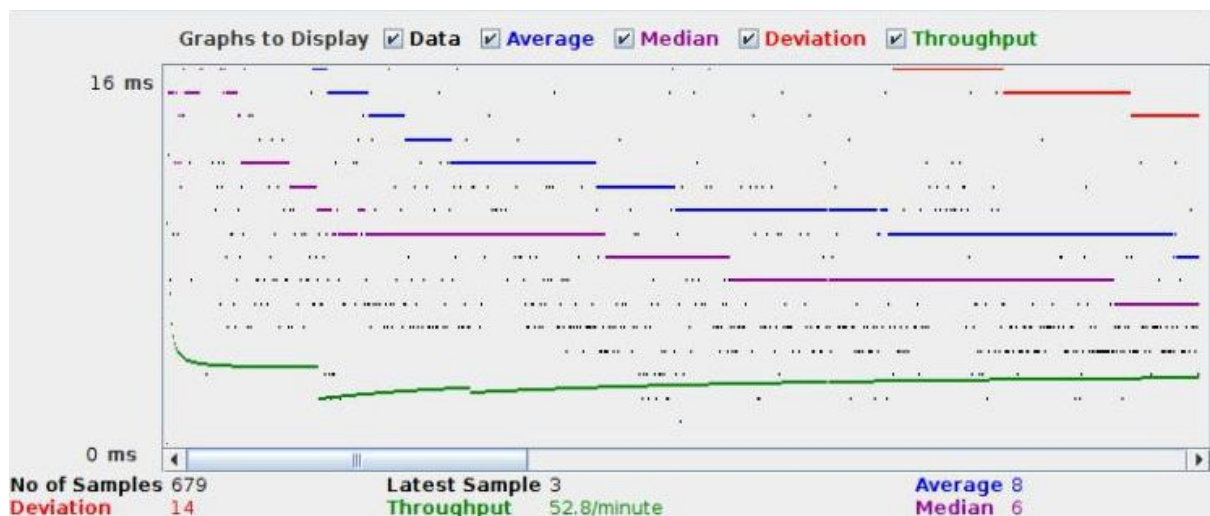
En la IP ponemos 192.168.58.105 para Ubuntu Server o 192.168.58.110 para CentOS.



Http Request->Add->Listener->Graph Results



Ahora veremos los resultados en Ubuntu Server:

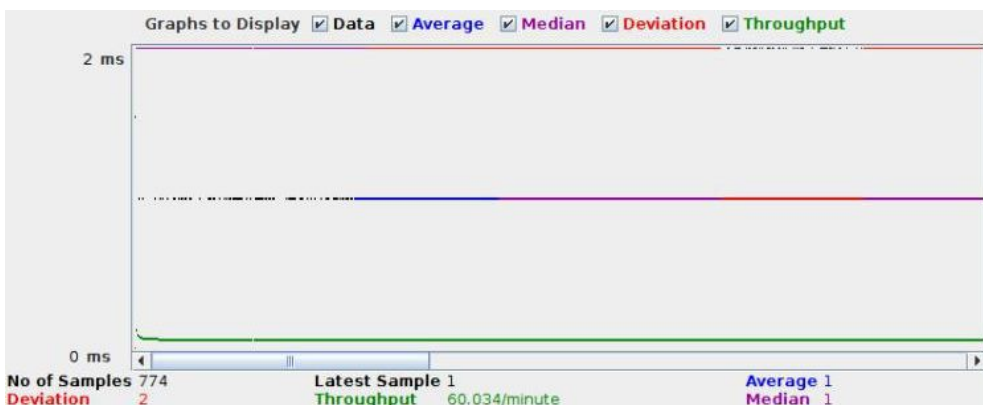


Ahora en /etc/apache2/apache2.conf cambiaremos los siguientes valores:

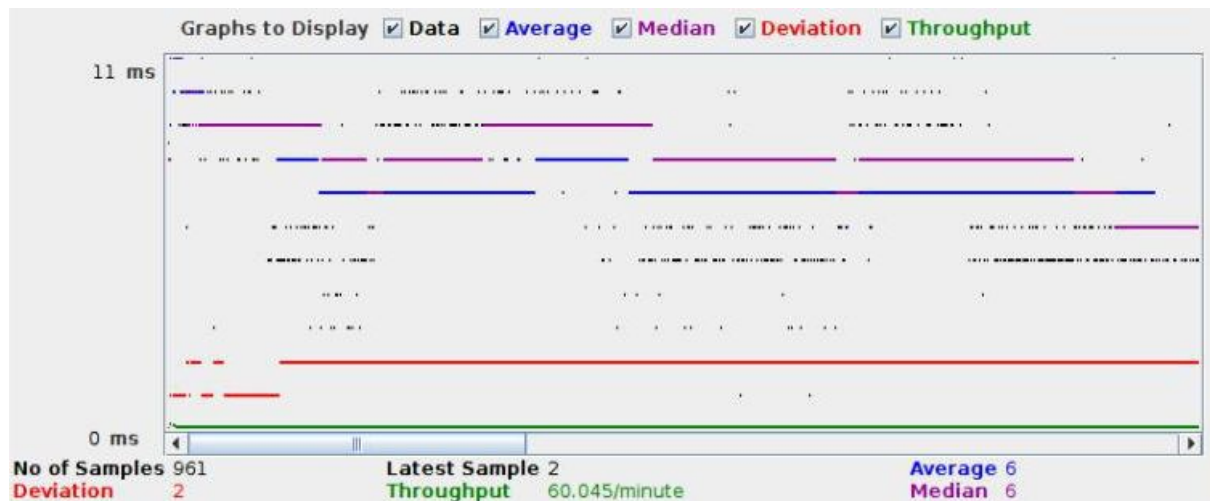
MaxKeepAliveRequest 30

KeepAliveTimeout 2

Y nos sale el siguiente resultado:



Ahora haremos lo mismo con CentOS:



Ahora en /etc/httpd/conf/httpd.conf cambiaremos :  
KeepAliveTimeout 20



Los valores que hemos cambiado significan:

MaxKeepAliveRequest: establece el número máximo de peticiones que admite el servidor por cada conexión persistente.

KeepAliveTimeout: establece el número de segundos que el servidor va a esperar después de haber dado servicio a una determinada petición.

## Bibliografía

- <https://www.ostechnix.com/phoronix-test-suite-open-source-testing-benchmarking-tool/>
- <https://httpd.apache.org/docs/2.4/programs/ab.html>
- <https://blog.diacode.com/testeando-el-rendimiento-de-tu-aplicacion-con-apache-bench>
- <http://jmeter.apache.org/usermanual/build-web-test-plan.html>
- <https://devops.profitbricks.com/tutorials/optimize-apache-performance-on-centos-7-1/>