

**Ingeniería de Servidores (2014-2015)**

Grado en Ingeniería Informática

Universidad de Granada

---

## Memoria Práctica 4

---

M<sup>a</sup> Cristina Heredia Gómez

16 de diciembre de 2014

## Índice

1. Cuestión 1: Instale la aplicación. ¿Qué comando permite listar los benchmarks disponibles?	3
2. Cuestión opcional 1 : Seleccione, instale y ejecute uno, comente los resultados. Atención: no es lo mismo un benchmark que una suite, instale un benchmark.	4
3. Cuestión 2 : De los parámetros que le podemos pasar al comando(ab) ¿Qué significa -c 30 ? ¿y -n 1000?	7
4. Cuestión 3 : Ejecute ab contra a las tres máquinas virtuales (desde el SO anfitrión a las máquinas virtuales de la red local) una a una (arrancadas por separado) y muestre las estadísticas. ¿Cuál es la que proporciona mejores resultados? Fíjese en el número de bytes transferidos¿es igual para cada máquina?	7
5. Cuestión opcional 3 : Lea el artículo y elabore un breve resumen.	10
6. Cuestión 4 : Instale y siga el tutorial en <a href="http://jmeter.apache.org/usermanual/build-web-test-plan.html">http://jmeter.apache.org/usermanual/build-web-test-plan.html</a> realizando capturas de pantalla y comentándolas. En vez de usar la web de jmeter, haga el experimento usando alguna de sus máquinas virtuales (Puede hacer una página sencilla, usar las páginas de phpmyadmin, instalar un CMS, etc.).	11
7. Cuestión 5 : Programe un benchmark usando el lenguaje que desee. El benchmark debe incluir: 1) Objetivo del benchmark 2) Métricas (unidades, variables, puntuaciones, etc.) 3) Instrucciones para su uso 4) Ejemplo de uso analizando los resultados	16

## Índice de figuras

1.1. ejecutando phoronix-test-suite . . . . .	3
1.2. mostrar benchmarks disponibles phoronix-test-suite . . . . .	4
2.1. sudokut ha sido instalado . . . . .	5
2.2. ejecutando benchmark sudokut . . . . .	5
2.3. resultado de ejecutar benchmark sudokut . . . . .	6
2.4. resultado de ejecutar benchmark sudokut . . . . .	6
2.5. resultado de ejecutar benchmark sudokut . . . . .	6
4.1. resultado de ejecutar ab . . . . .	8
4.2. resultado de ejecutar ab . . . . .	9
4.3. resultado de ejecutar ab . . . . .	10
6.1. creando grupo de hilos . . . . .	11
6.2. creando opciones por defecto para consultas HTTP . . . . .	12
6.3. creando consultas HTTP . . . . .	12
6.4. creando consultas HTTP . . . . .	13

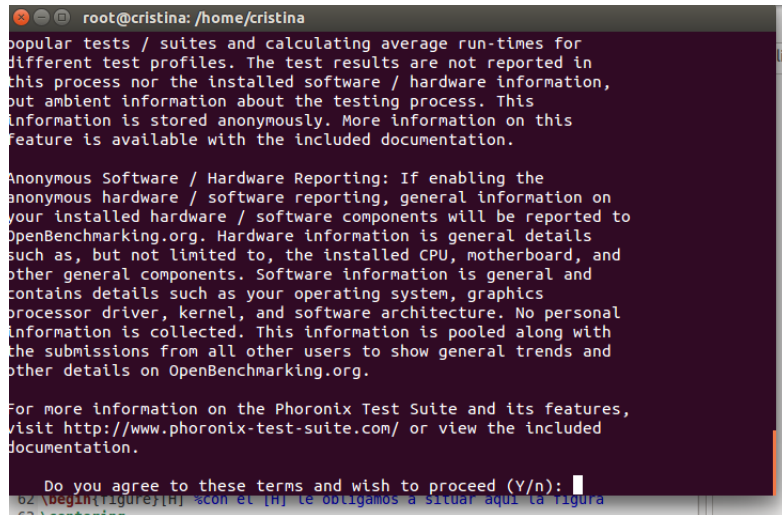
6.5. añadiendo gráfico de resultados . . . . .	13
6.6. ejecutando jmeter . . . . .	14
6.7. ejecutando jmeter . . . . .	15
7.1. benchmarkCasero.c . . . . .	16
7.2. ejemplo de ejecución de benchmarkCasero.c . . . . .	17

## Índice de tablas

## 1. Cuestión 1: Instale la aplicación. ¿Qué comando permite listar los benchmarks disponibles?

En ubuntu, se instala con: `apt-get install phoronix-test-suite`

Al ejecutarlo, nos pregunta si aceptamos la licencia y si permitimos enviar a los desarrolladores estadísticas anónimas para mejorar el programa:



```
root@cristina: /home/cristina
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.

Anonymous Software / Hardware Reporting: If enabling the
anonymous hardware / software reporting, general information on
your installed hardware / software components will be reported to
OpenBenchmarking.org. Hardware information is general details
such as, but not limited to, the installed CPU, motherboard, and
other general components. Software information is general and
contains details such as your operating system, graphics
processor driver, kernel, and software architecture. No personal
information is collected. This information is pooled along with
the submissions from all other users to show general trends and
other details on OpenBenchmarking.org.

For more information on the Phoronix Test Suite and its features,
visit http://www.phoronix-test-suite.com/ or view the included
documentation.

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

Figura 1.1: ejecutando phoronix-test-suite

Para listar los benchmarks disponibles:

`phoronix-test-suite list-available-tests`

y para listar las suites disponibles:

`phoronix-test-suite list-available-suites`

```
cris: /home/cristina
root@cristina: /home/cristina# phoronix-test-suite list-available-tests

Phoronix Test Suite v4.8.3
Available Tests

pts/aio-stress          - AIO-Stress          Disk
pts/apache             - Apache Benchmark    System
pts/apitest            - APITest             Graphics
pts/apitrace           - APITrace            Graphics
pts/battery-power-usage - Battery Power Usage System
pts/blake2             - BLAKE2              Processor
pts/blogbench          - BlogBench           Disk
pts/bork               - Bork File Encrypter Processor
pts/botan              - Botan               Processor
pts/build-apache        - Timed Apache Compilation Processor
pts/build-firefox       - Timed Firefox Compilation Processor
pts/build-imagemagick   - Timed ImageMagick Compilation Processor
pts/build-linux-kernel - Timed Linux Kernel Compilation Processor
pts/build-mplayer       - Timed MPlayer Compilation Processor
pts/build-php           - Timed PHP Compilation Processor
pts/build-webkitgtk     - Timed WebKitGTK Compilation Processor
pts/bullet             - Bullet Physics Engine Processor
pts/byte               - BYTE Unix Benchmark Processor
pts/c-ray              - C-Ray               Processor
pts/cachebench         - CacheBench          Processor
pts/cairo-demos        - Cairo Performance Demos Graphics
pts/cairo-perf-trace    - cairo-perf-trace    Graphics
pts/clomp              - CLOMP               Processor
pts/compilebench       - Compile Bench       Disk
pts/compress-7zip       - 7-Zip Compression   Processor
pts/compress-gzip       - Gzip Compression    Processor
pts/compress-lzma       - LZMA Compression    Processor
pts/compress-pbzip2     - Parallel BZIP2 Compression Processor
pts/corebreach         - CoreBreach          Graphics
pts/crafty             - Crafty              Processor
pts/csgo               - Counter-Strike: Global Offensive Graphics
pts/cstrike            - Counter-Strike Source Graphics
pts/cyclictst          - Cyclictst           System
pts/dbench             - Dbench              Disk
pts/dcrw               - dcrw                Processor
pts/dolfyn             - Dolfyn              Processor
```

Figura 1.2: mostrar benchmarks disponibles phoronix-test-suite

## 2. Cuestión opcional 1 : Seleccione, instale y ejecute uno, comente los resultados. Atención: no es lo mismo un benchmark que una suite, instale un benchmark.

Yo he elegido sudokut. En primer lugar lo instalo con: `phoronix-test-suite install sudokut`

```

cristina:/home/cristina
Creando árbol de dependencias...
Leyendo la información de estado...
unzip ya está en su versión más reciente.
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
libmemcached10 libmemcachedutil2
Use 'apt-get autoremove' to remove them.
Se instalarán los siguientes paquetes NUEVOS:
  mesa-utils
0 actualizados, 1 se instalarán, 0 para eliminar y 29 no actualizados.
Necesito descargar 34,4 kB de archivos.
Se utilizarán 133 kB de espacio de disco adicional después de esta operación.
Des:1 http://es.archive.ubuntu.com/ubuntu/ trusty/universe mesa-utils amd64 8.1.0-2 [34,4 kB]
Descargados 34,4 kB en 0seg. (36,0 kB/s)
Seleccionando el paquete mesa-utils previamente no seleccionado.
(Leyendo la base de datos ... 353002 ficheros o directorios instalados actualmente.)
Preparing to unpack .../mesa-utils_8.1.0-2_amd64.deb ...
Unpacking mesa-utils (8.1.0-2) ...
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...
Configurando mesa-utils (8.1.0-2) ...

Phoronix Test Suite v4.8.3

  To Install: pts/sudokut-1.0.0

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

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

  pts/sudokut-1.0.0:
    Test Installation 1 of 1
    1 File Needed [0.02 MB]
    Downloading: sudokut0.4-1.tar.bz2 [0.02MB]
    Downloading .....
    Installation Size: 0.1 MB
    Installing Test @ 00:33:52

```

Figura 2.1: sudokut ha sido instalado

y lo probamos:

```

root@cristina:/home/cristina# phoronix-test-suite benchmark sudokut

Phoronix Test Suite v4.8.3

  Installed: pts/sudokut-1.0.0

System Information

Hardware:
Processor: Intel Core i5-2430M @ 2.40GHz (4 Cores), Motherboard: Sony VAIO, Chipset: Intel 2nd Generation Core Family DRAM, Memory: 1 x 4096
DDR3, Disk: 750GB Western Digital WD7500BPVT-5, Graphics: NVIDIA GeForce 410M 1024MB, Audio: Conexant CX20590, Network: Realtek RTL8111/8168
1 + Qualcomm Atheros AR9285 Wireless

Software:
OS: Ubuntu 14.04, Kernel: 3.13.0-37-generic (x86_64), Desktop: Unity 7.2.3, Display Server: X Server 1.15.1, Display Driver: nouveau 1.0.10,
nGL: 3.3 Mesa 10.1.3 Gallium 0.4, Compiler: GCC 4.8, File-System: ext4, Screen Resolution: 1366x768

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

```

Figura 2.2: ejecutando benchmark sudokut

```

Current Description: Intel Core i5-2430M testing with a Sony VAIO and NVIDIA GeForce 410M 1024MB on Ubuntu 14.04 via the Phoronix Test Suite.
New Description: cuestion opcional1

Sudoku 0.4:
pts/sudoku-1.0.0
Test 1 of 1
Estimated Trial Run Count: 3
Estimated Time To Completion: 3 Minutes
Started Run 1 @ 00:39:23
Started Run 2 @ 00:39:53
Started Run 3 @ 00:40:19 [Std. Dev: 1.84%]

Test Results:
23.513960123062
24.009920120239
23.147936820984

Average: 23.56 Seconds

```

Figura 2.3: resultado de ejecutar benchmark sudoku

visualizando estos resultados en el navegador...:


cris	
	ise
Phoronix Test Suite	
Sudoku	23.56
Standard Error	0.25
Standard Deviation	1.84%
PHORONIX-TEST-SUITE.COM	

Figura 2.4: resultado de ejecutar benchmark sudoku

## Test Results

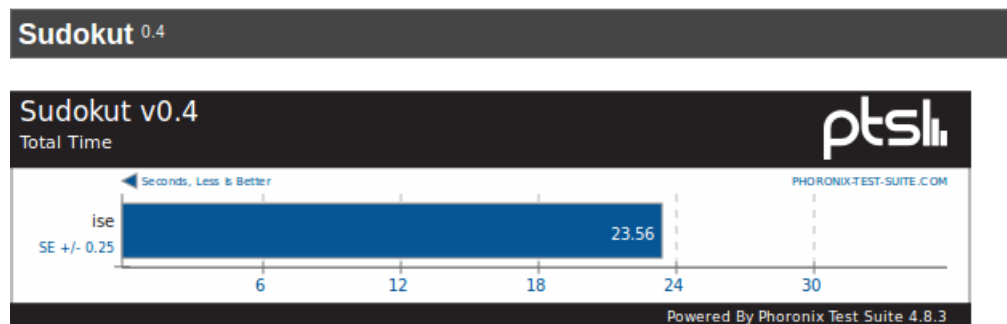


Figura 2.5: resultado de ejecutar benchmark sudoku

Donde vemos que 23.56 segundos es el tiempo que tarda mi ordenador en resolver un total de 100 sudokus.

### **3. Cuestión 2 : De los parámetros que le podemos pasar al comando(ab) ¿Qué significa -c 30 ? ¿y -n 1000?**

La opción -c especifica la concurrencia. Con ella se puede indicar el número máximo de peticiones que se podrán ejecutar simultáneamente, por ejemplo con -c 30 lo estamos poniendo a 30.

[mana]

La opción -n especifica las request(peticiones).Con ella se puede indicar el número de peticiones de página web que se harán al servidor.Por ejemplo, con -n 1000 estamos estableciendo que se hagan mil peticiones. [manb]

### **4. Cuestión 3 : Ejecute ab contra a las tres máquinas virtuales (desde el SO anfitrión a las máquina virtuales de la red local) una a una (arrancadas por separado) y muestre las estadísticas. ¿Cuál es la que proporciona mejores resultados? Fíjese en el número de bytes transferidos¿es igual para cada máquina?**

En ubuntu: ejecutamos ab -c 30 -n 1000 localhost/



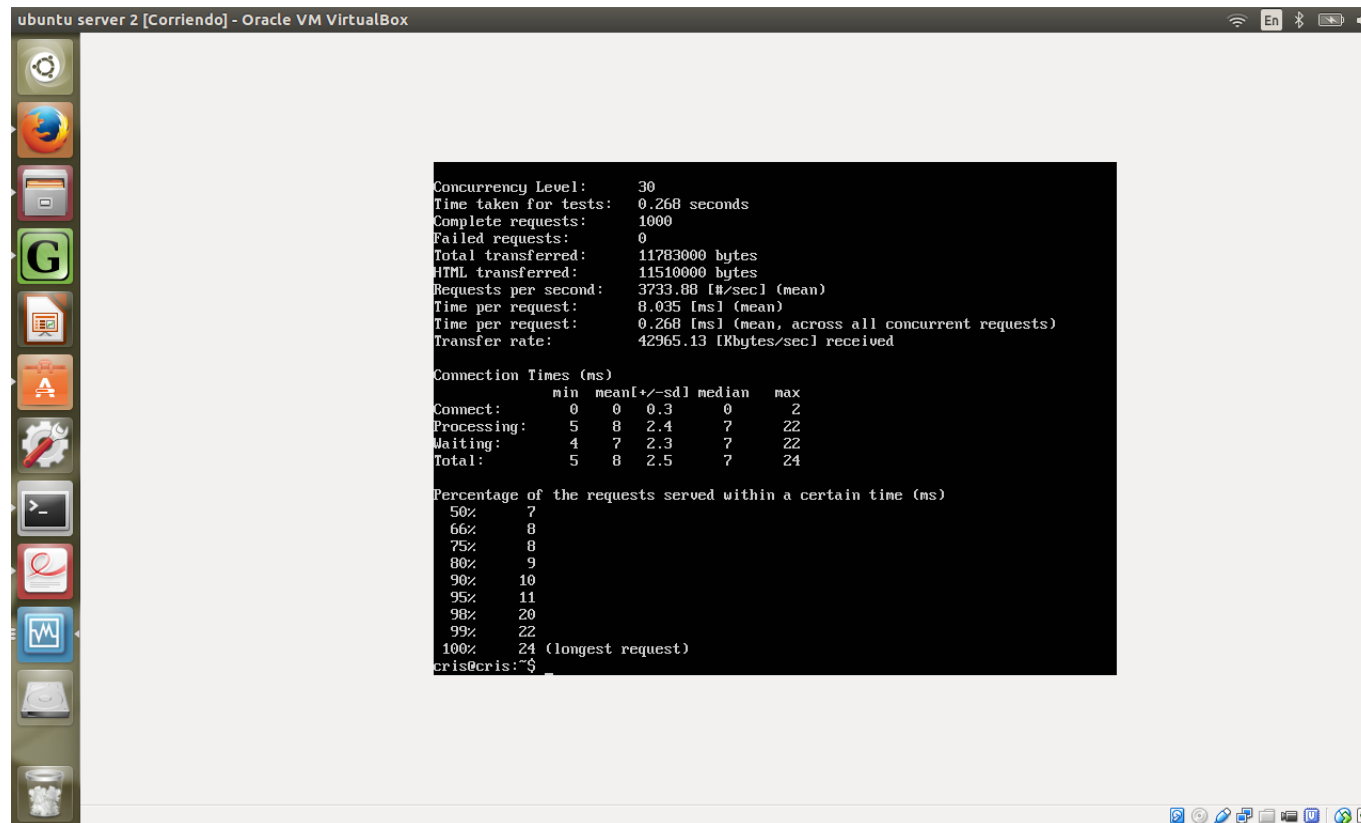
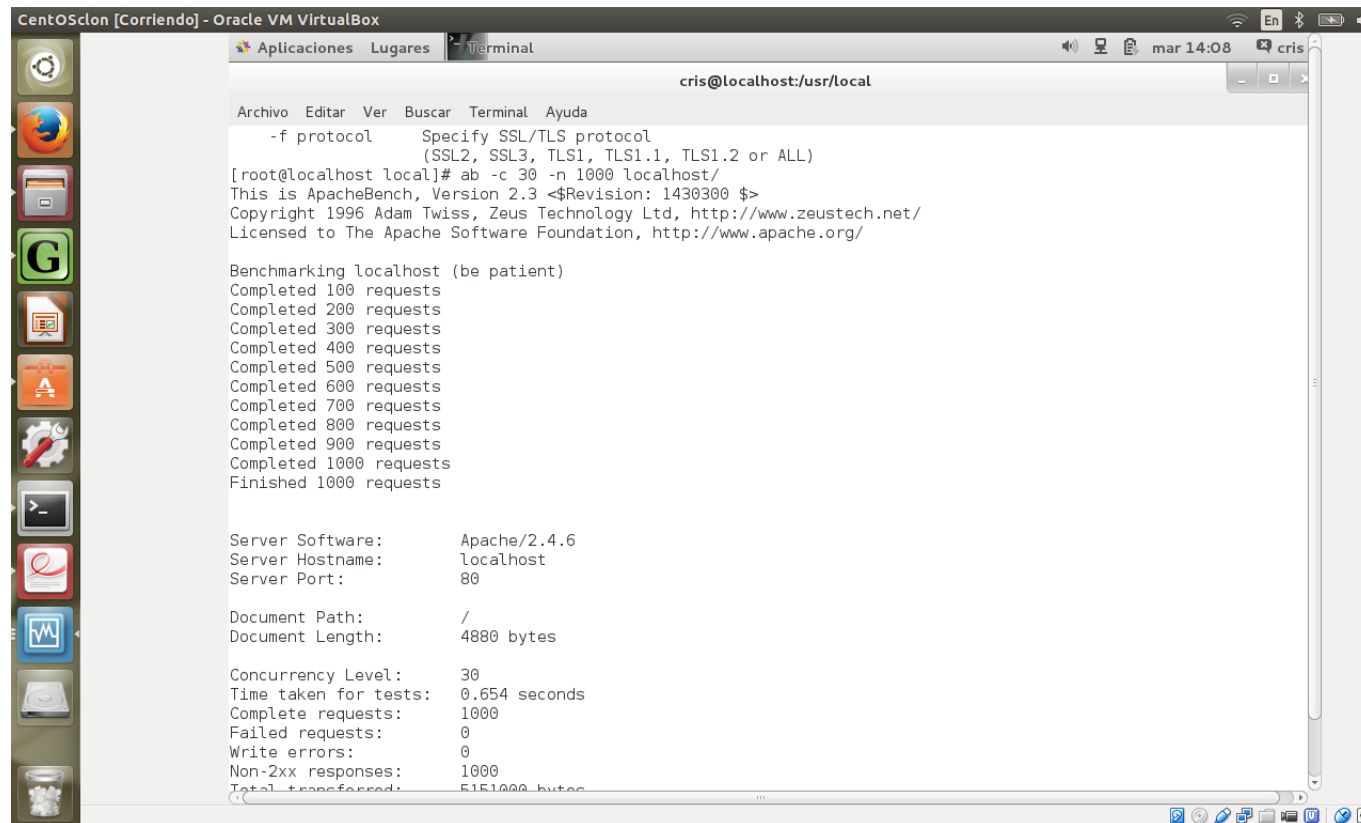


Figura 4.1: resultado de ejecutar ab

En CentOS:



```
CentOSclon [Corriendo] - Oracle VM VirtualBox
Aplicaciones Lugares Terminal
cris@localhost:/usr/local

Archivo Editar Ver Buscar Terminal Ayuda
-f protocol Specify SSL/TLS protocol
(SSL2, SSL3, TLS1, TLS1.1, TLS1.2 or ALL)
[root@localhost local]# ab -c 30 -n 1000 localhost/
This is ApacheBench, Version 2.3 <$Revision: 1430300 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

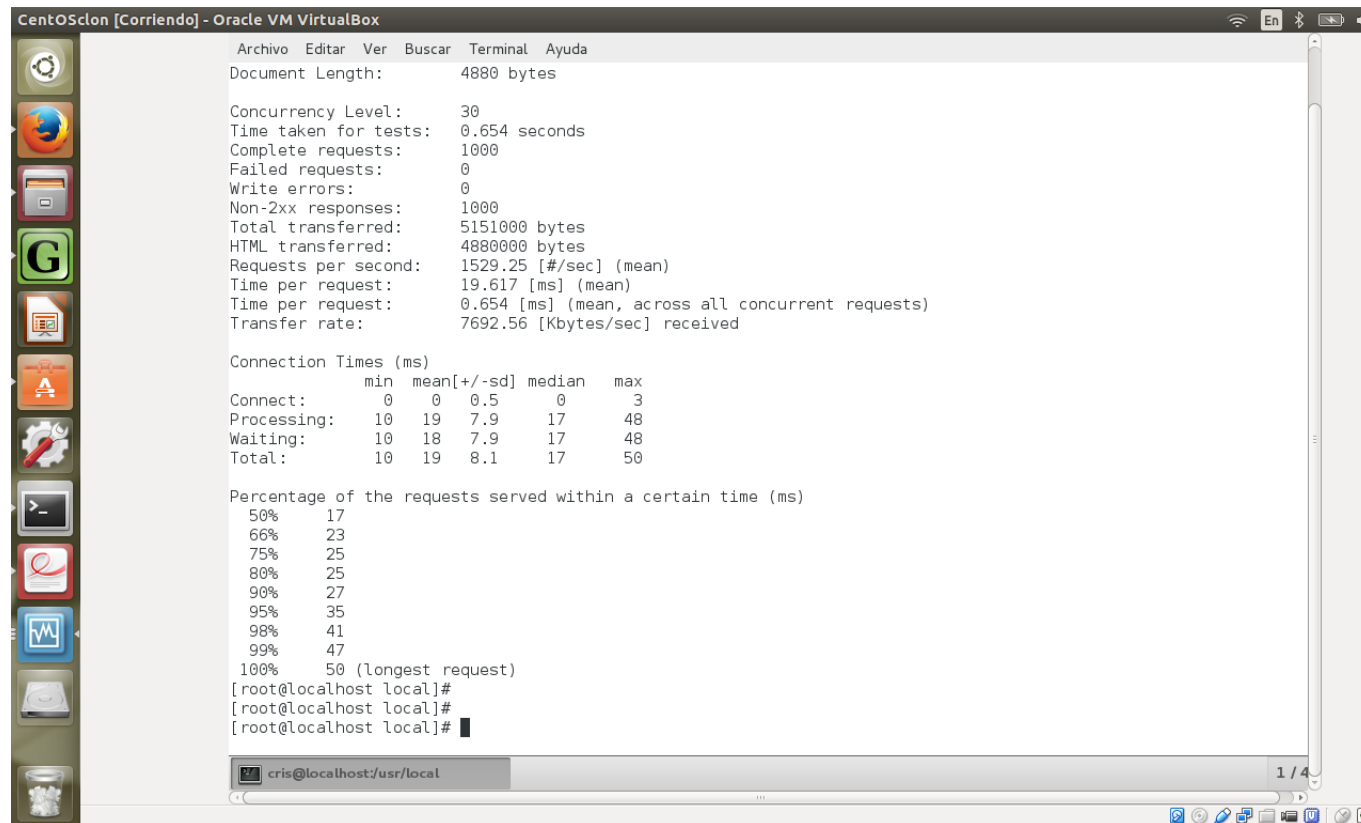
Benchmarking localhost (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests

Server Software: Apache/2.4.6
Server Hostname: localhost
Server Port: 80

Document Path: /
Document Length: 4880 bytes

Concurrency Level: 30
Time taken for tests: 0.654 seconds
Complete requests: 1000
Failed requests: 0
Write errors: 0
Non-2xx responses: 1000
Total transferred: 5151000 bytes
```

Figura 4.2: resultado de ejecutar ab



```
CentOSclon [Corriendo] - Oracle VM VirtualBox
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
Document Length:      4880 bytes

Concurrency Level:      30
Time taken for tests:    0.654 seconds
Complete requests:      1000
Failed requests:         0
Write errors:           0
Non-2xx responses:      1000
Total transferred:      5151000 bytes
HTML transferred:       4880000 bytes
Requests per second:    1529.25 [#/sec] (mean)
Time per request:       19.617 [ms] (mean)
Time per request:       0.654 [ms] (mean, across all concurrent requests)
Transfer rate:          7692.56 [Kbytes/sec] received

Connection Times (ms)
      min     mean[+/-sd] median   max
Connect:    0       0   0.5      0      3
Processing: 10      19   7.9     17     48
Waiting:    10      18   7.9     17     48
Total:       10      19   8.1     17     50

Percentage of the requests served within a certain time (ms)
 50%      17
 66%      23
 75%      25
 80%      25
 90%      27
 95%      35
 98%      41
 99%      47
100%      50 (longest request)
[root@localhost local]#
[root@localhost local]#
[root@localhost local]#
```

Figura 4.3: resultado de ejecutar ab

Observando los resultados, vemos que en CentOS el número medio de respuestas por segundo es de 1529.25 mientras que en Ubuntu Server es de 3733.88. El número de bytes transferidos observamos que también difiere en cada máquina, pues en CentOS es de 7692.56 kbytes/sec mientras que en Ubuntu es de 42965.13 kbytes/sec (observamos que en Ubuntu el número de bytes transferidos es mayor). Por último vemos que el 100 % de las respuestas se sirven en 50ms en CentOS mientras que en Ubuntu Server se tarda menos en servirlos(24ms). Concluyo entonces, que Ubuntu da mejores resultados.

## 5. Cuestión opcional 3 : Lea el artículo y elabore un breve resumen.

¿Es Gatling mejor que JMeter, o viceversa?

Para responder a esta pregunta, Tim Koopmans nos cuenta los resultados obtenidos en una prueba que efectuaron sobre ambos. Para llevar a cabo esta prueba, usaron Flood.io, una plataforma de pruebas de carga distribuida, probando con un solo nodo.

Establecieron los valores de concurrencia a 10.000 usuarios y las peticiones a 30.000 por minuto. ¿Cuáles fueron los resultados?

Pues finalmente dedujeron que no había mucha diferencia entre ambas herramientas, pues daban resultados muy similares, a pesar de algunas diferencias entre ellas como que JMeter tiene recursos más pesados en cuanto a JVM, procesador y memoria que Gatling.

En cuanto a la concurrencia, podemos decir que Gatling tiene algunas limitaciones en la capacidad de registrar con precisión la carga en bytes, mientras que JMeter parece que no tiene problema a la hora de gestionar la carga cuando se ejecuta con una asignación de memoria adecuada.

[flo]

## 6. Cuestión 4 : Instale y siga el tutorial en <http://jmeter.apache.org/usermanual/build-web-test-plan.html> realizando capturas de pantalla y comentándolas. En vez de usar la web de jmeter, haga el experimento usando alguna de sus máquinas virtuales (Puede hacer una página sencilla, usar las páginas de phpmyadmin, instalar un CMS, etc.).

En primer lugar lo instalamos, con: `sudo apt-get install jmeter`

Una vez instalado, lo ejecutamos y seleccionamos "Plan de Pruebas" -> "Añadir" -> "Grupo de Hilos":

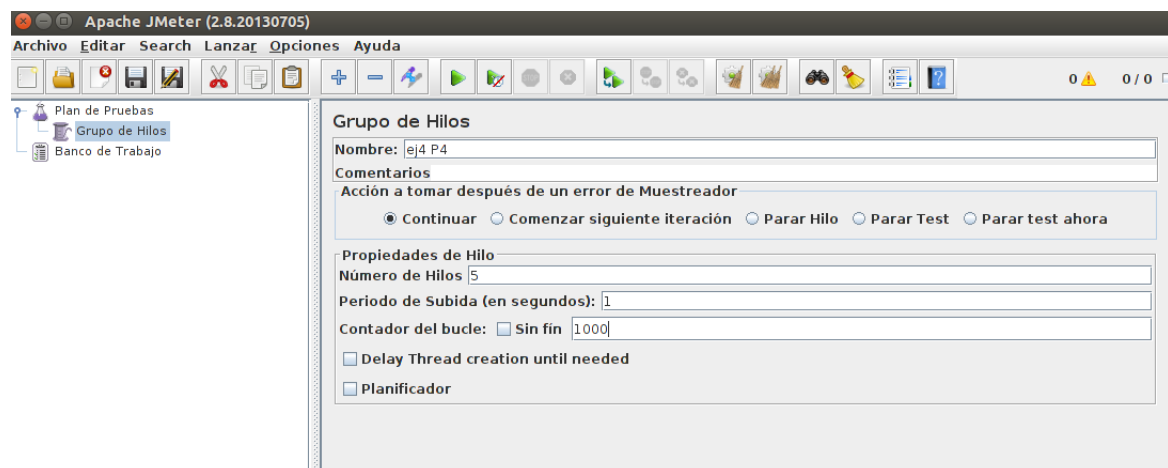


Figura 6.1: creando grupo de hilos

Ahora creamos las opciones por defecto para consultas HTTP. Esto lo hacemos con: Añadir -> Elementos de Configuración -> Valores por defecto para Petición HTTP:

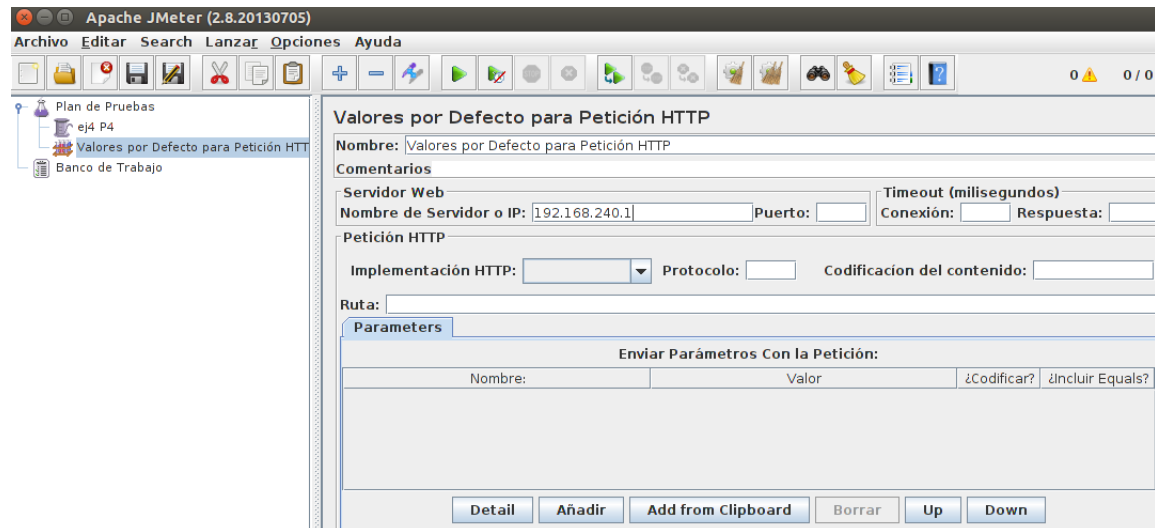


Figura 6.2: creando opciones por defecto para consultas HTTP

creamos dos consultas HTTP. Una que nos lleve al directorio raíz y otra a una página con imágenes. Lo hacemos en: Añadir->Muestreador->petición HTTP :

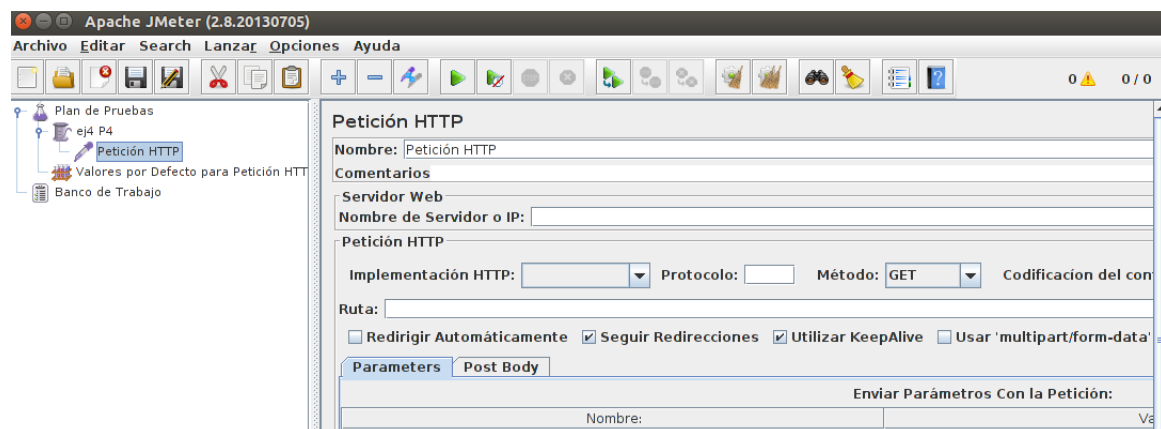


Figura 6.3: creando consultas HTTP

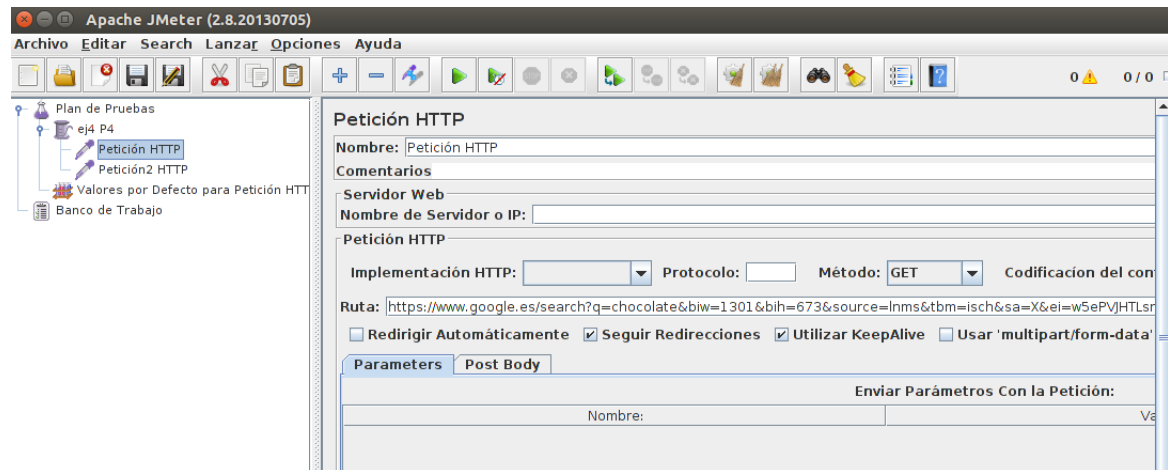


Figura 6.4: creando consultas HTTP

Por último, añadimos el gráfico de Resultados: seleccionaos Añadir->receptor->Gráfico de resultados

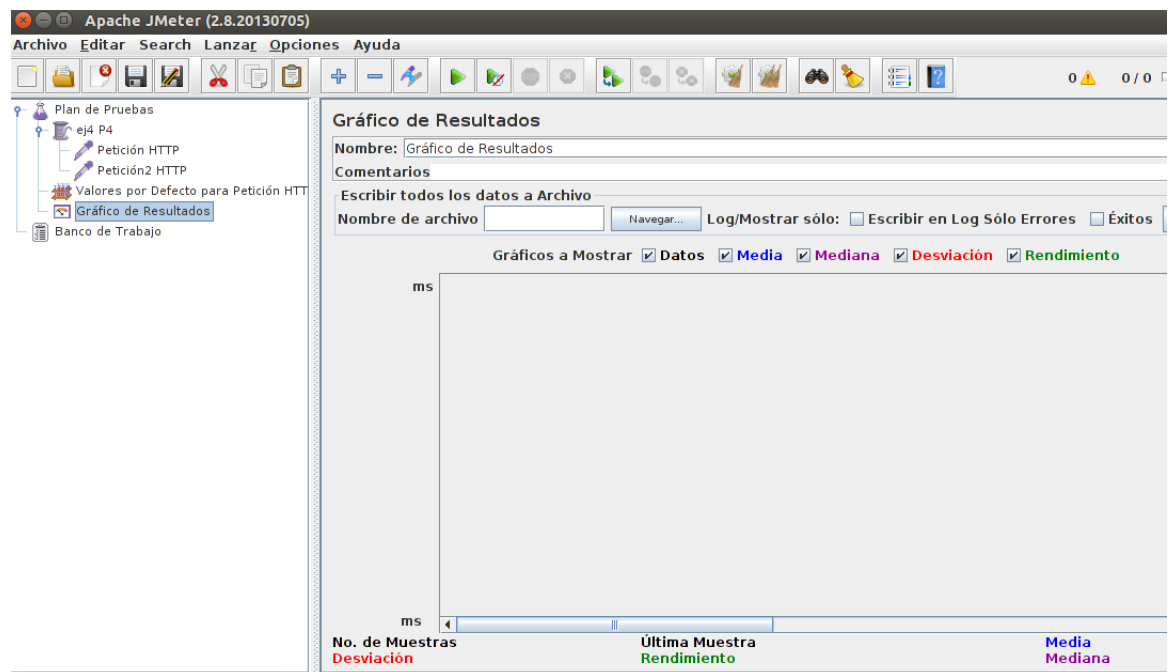


Figura 6.5: añadiendo gráfico de resultados

y ya pulsamos: lanzar->arrancar:

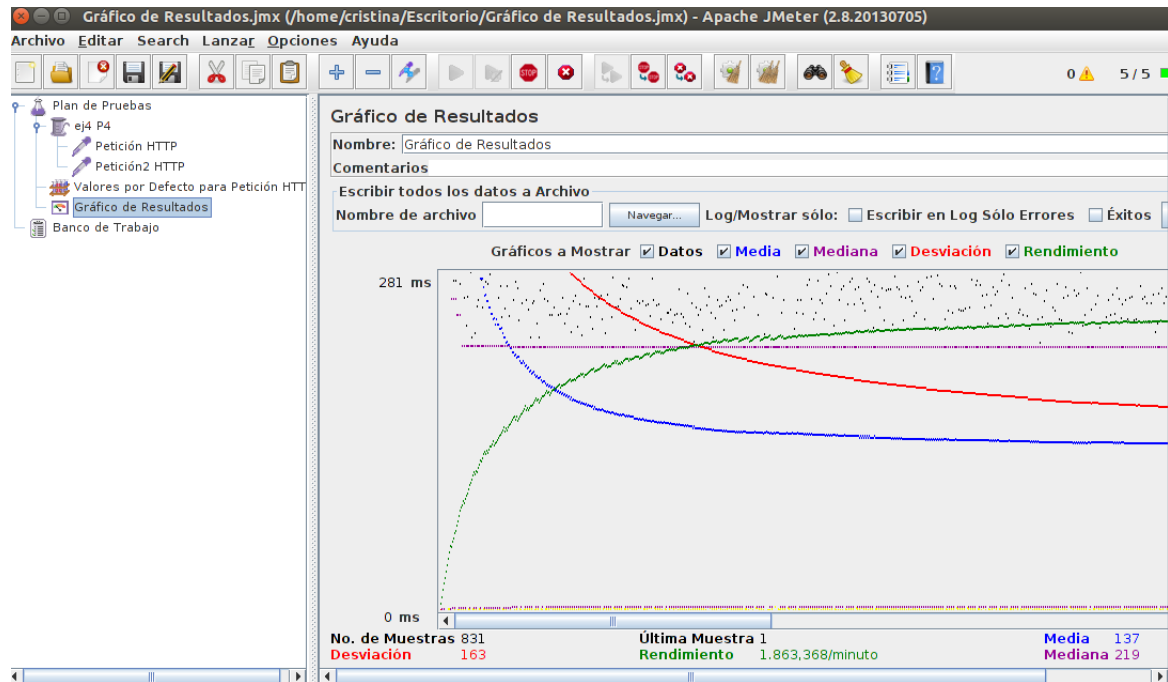


Figura 6.6: ejecutando jmeter

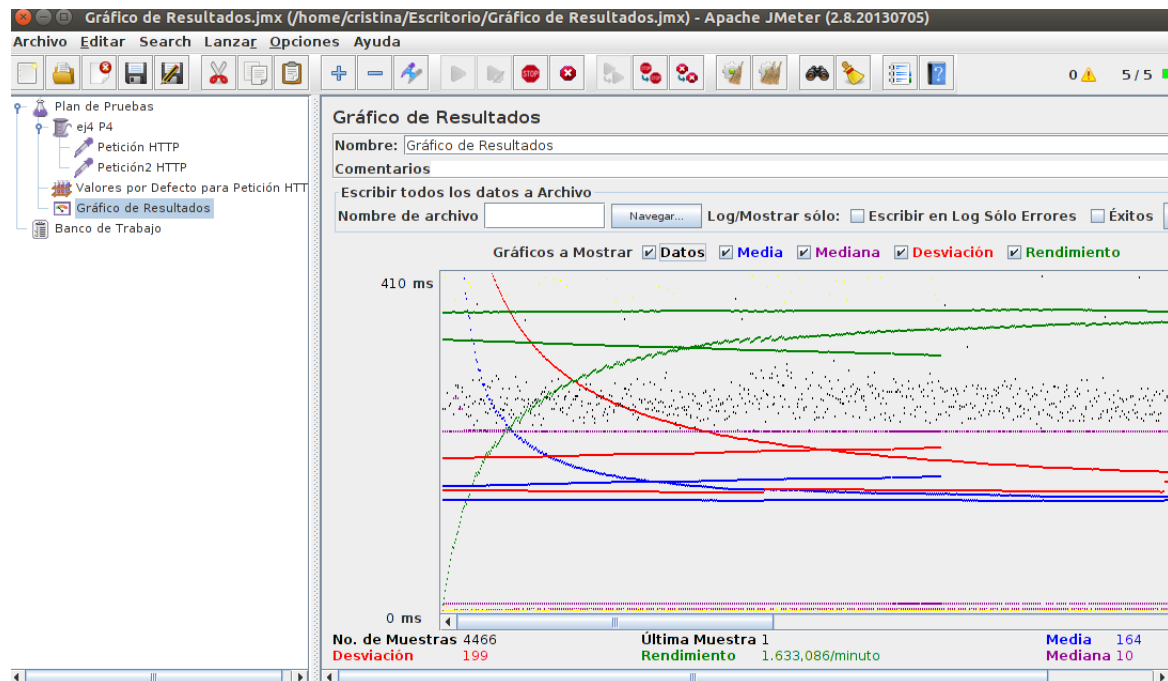


Figura 6.7: ejecutando jmeter

Vemos que el rendimiento comienza siendo bajo, al contrario que la desviación (pues al parecer son funciones inversas, o al menos en este caso lo parecen), sin embargo conforme se van analizando más datos, vemos que la desviación y la media se estabilizan y la mediana disminuye.



7. **Cuestión 5 : Programe un benchmark usando el lenguaje que desee. El benchmark debe incluir: 1) Objetivo del benchmark 2) Métricas (unidades, variables, puntuaciones, etc.) 3) Instrucciones para su uso 4) Ejemplo de uso analizando los resultados**

```
benchmarkCasero.c x
/* Cristina Heredia */
/*benchmark casero para medir tiempo de procesamiento de la cpu */
#include <sys/time.h>
#include <math.h>
#include<stdio.h>

void benchmark(){
int i,j=0;
float l,k=1.0;

for( i=0; i<10000;i++){
    for( j=0; j<1000;j++){
        k=k*69 + k*206 +k* 100;
        for(l=1;l<1000;l++){
            k*l;
        }
    }
}
k*=k;
}|
int main(int argc, char **argv){

struct timeval timeB;
struct timeval timeE;

gettimeofday(&timeB,NULL);
benchmark();
gettimeofday(&timeE,NULL);

double startTime = timeB.tv_sec*1000000 + (timeB.tv_usec);
double endTime= timeE.tv_sec*1000000 + (timeE.tv_usec);
double ProcTime=endTime-startTime;
printf("%s","tiempo de procesamiento:");
printf("%f\n",ProcTime);
return(0);
}
```

Figura 7.1: benchmarkCasero.c

ejemplo de ejecución del benchmark casero en mi pc, con 4 cpus y 4G de RAM:

```
cristina@cristina:~$ gcc -o benchmark benchmarkCasero.c
cristina@cristina:~$ ./benchmark
tiempo de procesamiento:33510939.000000
cristina@cristina:~$
```

Figura 7.2: ejemplo de ejecución de benchmarkCasero.c

[www] [rab]  
\*Bibliografía

## Referencias

- [flo] flood.io/blog. *jmeter-and-gatling*. <https://flood.io/blog/11-benchmarking-jmeter-and-gatling>.
- [mana] linux man. *Manual page ab(1) line 1*.
- [manb] linux man. *Manual page ab(1) line 29*.
- [rab] rabbit.eng.miami.edu. *timeofday*. <http://rabbit.eng.miami.edu/info/functions/time.html>.
- [www] www.codingunit.com. *printf*. <http://www.codingunit.com/printf-format-specifiers-format-conversions-and-formatted-output>.