

INGENIERÍA DE SERVIDORES (2016-2017)
GRADO EN INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE GRANADA

Memoria Práctica 3

Cristian Vélez Ruiz

6 de diciembre de 2016

Índice

1. Cuestión 1	4
2. Cuestión 2	4
3. Cuestión 3	6
4. Cuestión 4	8
5. Cuestión 5	11
6. Cuestión 6	18
7. Cuestión 7	21
8. Cuestión 8	21
9. Cuestión 9	24

Índice de figuras

2.1. Creación crontab	5
2.2. Creación crontab 2	5
3.1. Usb sin conectar	6
3.2. Usb conectado	7
3.3. Diferencias entre archivos	7
4.1. Perfmon paso 1	8
4.2. Perfmon paso 2	8
4.3. Perfmon paso 3	9
4.4. Perfmon paso 4	9
4.5. Perfmon paso 5	10
4.6. Perfmon paso 6	10
5.1. Recopilador paso 1	11
5.2. Recopilador paso 2	12
5.3. Recopilador paso 3	12
5.4. Recopilador paso 4	13
5.5. Recopilador paso 5	13
5.6. Recopilador paso 6	14
5.7. Recopilador paso 7	14
5.8. Recopilador paso 8	15
5.9. Recopilador paso 9	15
5.10. Recopilador paso 10	16
5.11. Recopilador paso 11	17
5.12. Recopilador paso 12	17

5.13. Recopilador paso 13	18
6.1. Munin 1	19
6.2. Munin 2	19
6.3. Munin 3	20
6.4. Munin 4	20
8.1. Python 1	22
8.2. Python 2	22
8.3. Python 3	23
8.4. Python 4	23
8.5. Python 5	23
9.1. Profiler mysql 1	24
9.2. Profiler mysql 2	25
9.3. Profiler mysql 3	26
9.4. Profiler mysql 4	27
9.5. Profiler mysql 5	28
9.6. Profiler mysql 6	29
9.7. Profiler mysql 7	30
9.8. Profiler mysql 8	30

Índice de tablas

1. a) ¿Qué archivo le permite ver qué programas se han instalado con el gestor de paquetes? 1.b) ¿Qué significan las terminaciones. 1.gz o .2.gz de los archivos en ese directorio?

- (a) Para poder ver los programas que se han instalado con el gestor de paquetes podemos ir al archivo `/var/log/dpkg.log`, este fichero pertenece al log de dpkg [4], que es la base de nuestro gestor de paquetes.

Si queremos visualizar nosotros mismos los paquetes que están instalados

Otra opción para visualizar paquetes instalados en nuestro sistema es usar la orden `dpkg -get-selections`, nos muestra lo que está instalado en nuestro sistema.

- (b) Las terminaciones de esos archivos hacen referencia a que son archivos comprimidos, se crean para ahorrar espacio en el sistema sin tener que eliminar todos los log.

Cuando se va a generar un nuevo archivo comprimido se le cambia el nombre a los anteriores, por tanto el fichero mas antiguo en este caso sería el fichero `.2.gz`.

2. ¿Qué archivo ha de modificar para programar una tarea? Escriba la línea necesaria para ejecutar una vez al día una copia del directorio `/codigo` a `/seguridad/$fecha` donde `$fecha` es la fecha actual (puede usar el comando date)

Para poder programar una tarea es necesario la ayuda de cron que es el demonio que se encarga de ir ejecutando las tareas que se definen en el crontab [3], para programar una tarea en el crontab usaremos `crontab -e`, ya que no tenemos ningún crontab aun, en el caso de que existiera uno ya anteriormente realizaríamos lo mismo pero con `crontab -l`, para no tener que eliminar lo anterior.

Nuestro script es,

```
DIA='date +"%d/%m/%y" '
mkdir -p ~/seguridad/$DIA
cp -r ~/codigo/* ~/seguridad/$DIA
```

Con eso lo que conseguimos es hacer una variable con el nombre de la nueva carpeta en formato día/mes/año y a continuación lo que hacemos es copiar lo que tenemos en el directorio código a esa carpeta.

Ahora lo que tenemos que hacer es automatizarlo para que se ejecute una vez al día, como no tenemos ningún crontab creado anteriormente lanzaremos `crontab -e` y programaremos nuestro script.

The screenshot shows a terminal window titled "Terminal" running the "GNU nano 2.2.6" editor. The file being edited is "/tmp/crontab.7H0Cej/crontab". The status bar indicates the file is "Modificado" (modified). The content of the crontab is:

```
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 0 * * * ~/script.bash
```

At the bottom of the screen, there is a menu of keyboard shortcuts:

- ^G Ver ayuda
- ^O Guardar
- [Se ignora XOFF, mmh mmh]
- ^X Salir
- ^J Justificar
- ^R Leer fich.
- ^Y Pág. ant.
- ^K Cortar Texto
- ^C Posición
- ^W Buscar
- ^V Pág. sig.
- ^U PegarTxt
- ^T Ortografía

Figura 2.1: Creación crontab

Con esa configuración le decimos que se ejecute a la hora 0 y en el minuto 0 todos los días y nos ejecute el script anterior.

Guardamos y seguidamente nos informa de que se ha creado nuestro nuevo crontab.

The screenshot shows a terminal window with the following session:

```
(10:18:05)[CristianVelez--]$> crontab -e
crontab: installing new crontab
(10:21:52)[CristianVelez--]$>
```

Figura 2.2: Creación crontab 2

Otra manera sería meter el script en la carpeta `/etc/cron.daily/`, y cron [2] se encargará de ejecutarla diariamente.

3. **Pruebe a ejecutar el comando, conectar un dispositivo USB y vuelva a ejecutar el comando. Copie y pegue la salida del comando.(considere usar dmesg | tail). Comente qué observa en la información mostrada.**

Si ejecutamos dmesg -T nos mostrará el log del kernel añadiendo la fecha y hora para poder ubicar mejor las cosas en el tiempo, lo que voy a hacer es hacer una llamada a dmesg -T y lo guardaré en un fichero, a continuación voy a introducir un pendrive, esperaré unos segundos y volveré a repetir el proceso pero guardándolo en otro fichero, para obtener lo que ha ocasionado el USB, bastará con hacer un diff entre los dos archivos

Primero guardo en sinUSB la información antes de conectar el USB,

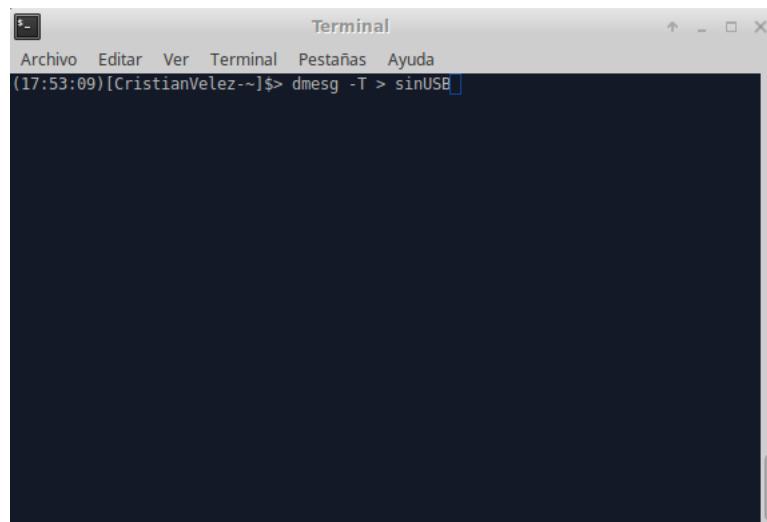
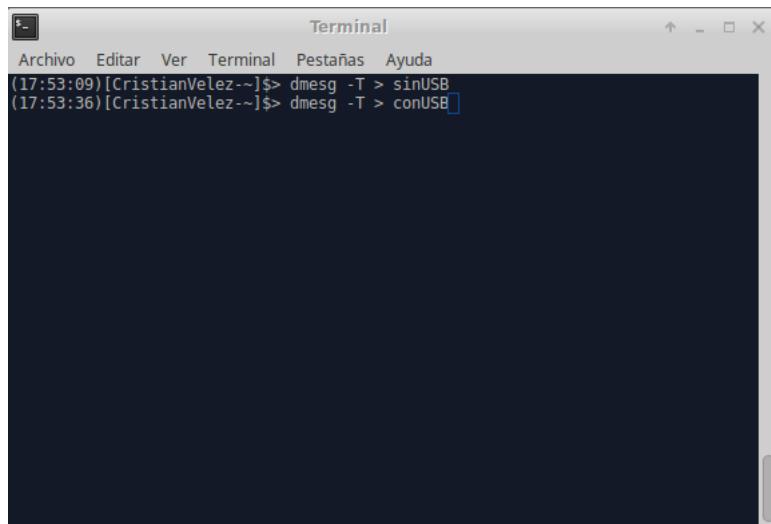


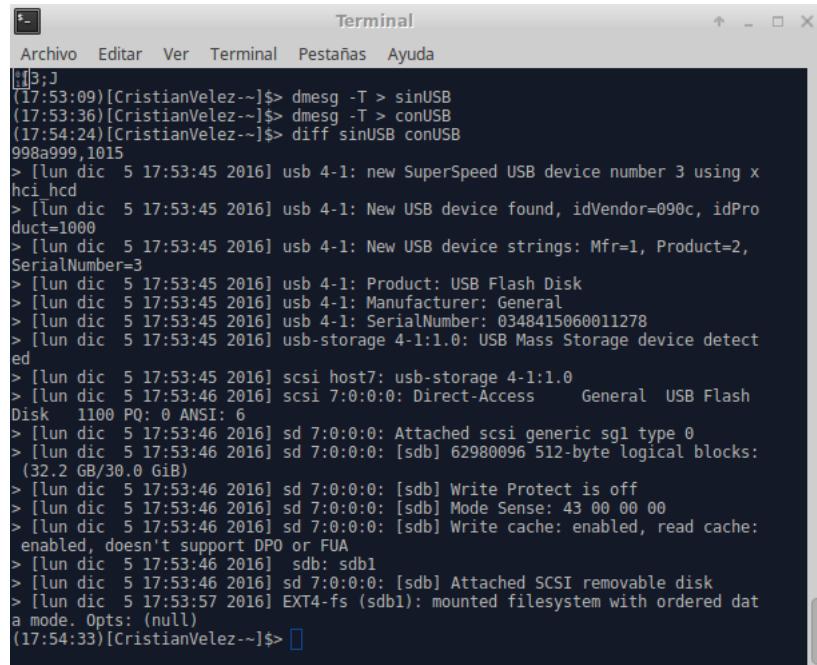
Figura 3.1: Usb sin conectar

Conecto el USB y espero unos segundos, y vuelvo a repetir el proceso anterior cambiando el archivo a conUSB,



```
Archivo Editar Ver Terminal Pestañas Ayuda
(17:53:09)[CristianVelez-~]$> dmesg -T > sinUSB
(17:53:36)[CristianVelez-~]$> dmesg -T > conUSB
```

Figura 3.2: Usb conectado



```
Archivo Editar Ver Terminal Pestañas Ayuda
[13:3]
(17:53:09)[CristianVelez-~]$> dmesg -T > sinUSB
(17:53:36)[CristianVelez-~]$> dmesg -T > conUSB
(17:54:24)[CristianVelez-~]$> diff sinUSB conUSB
998a999,1015
> [lun dic  5 17:53:45 2016] usb 4-1: new SuperSpeed USB device number 3 using x
hci_hcd
> [lun dic  5 17:53:45 2016] usb 4-1: New USB device found, idVendor=090c, idPro
duct=1000
> [lun dic  5 17:53:45 2016] usb 4-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
> [lun dic  5 17:53:45 2016] usb 4-1: Product: USB Flash Disk
> [lun dic  5 17:53:45 2016] usb 4-1: Manufacturer: General
> [lun dic  5 17:53:45 2016] usb 4-1: SerialNumber: 0348415060011278
> [lun dic  5 17:53:45 2016] usb-storage 4-1:1.0: USB Mass Storage device detect
ed
> [lun dic  5 17:53:45 2016] scsi host7: usb-storage 4-1:1.0
> [lun dic  5 17:53:46 2016] scsi 7:0:0:0: Direct-Access      General  USB Flash
Disk 1100 PQ: 0 ANSI: 6
> [lun dic  5 17:53:46 2016] sd 7:0:0:0: Attached scsi generic sg1 type 0
> [lun dic  5 17:53:46 2016] sd 7:0:0:0: [sdb] 62980096 512-byte logical blocks:
(32.2 GB/30.0 GiB)
> [lun dic  5 17:53:46 2016] sd 7:0:0:0: [sdb] Write Protect is off
> [lun dic  5 17:53:46 2016] sd 7:0:0:0: [sdb] Mode Sense: 43 00 00 00
> [lun dic  5 17:53:46 2016] sd 7:0:0:0: [sdb] Write cache: enabled, read cache:
enabled, doesn't support DPO or FUA
> [lun dic  5 17:53:46 2016] sdb: sdb1
> [lun dic  5 17:53:46 2016] sd 7:0:0:0: [sdb] Attached SCSI removable disk
> [lun dic  5 17:53:57 2016] EXT4-fs (sdb1): mounted filesystem with ordered dat
a mode. Opts: (null)
(17:54:33)[CristianVelez-~]$>
```

Figura 3.3: Diferencias entre archivos

Como se puede ver detecta que es un pendrive 3.0, así como la identidad de su vendedor y el número de serie del producto, también comprueba si está protegido frente a escritura y es solo de lectura, detecta las particiones que tiene y finalmente monta las particiones.

4. Ejecute el monitor de “System Performance” y muestre el resultado. Incluya capturas de pantalla comentando la información que aparece.

Para poder ejecutar perfmon [6], lo primero que debemos hacer es llamarlo desde ejecutar.

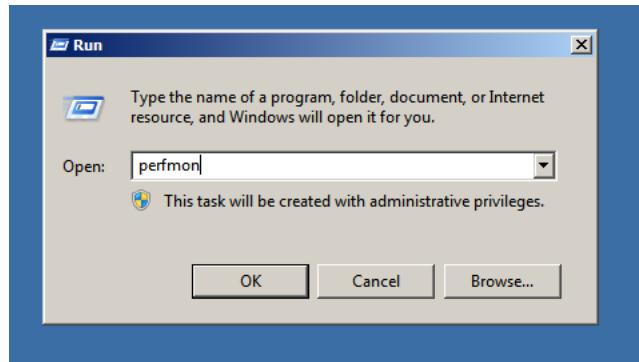


Figura 4.1: Perfmon paso 1

Seguidamente ya nos saldrá la ventana del *Perfomance Monitor*

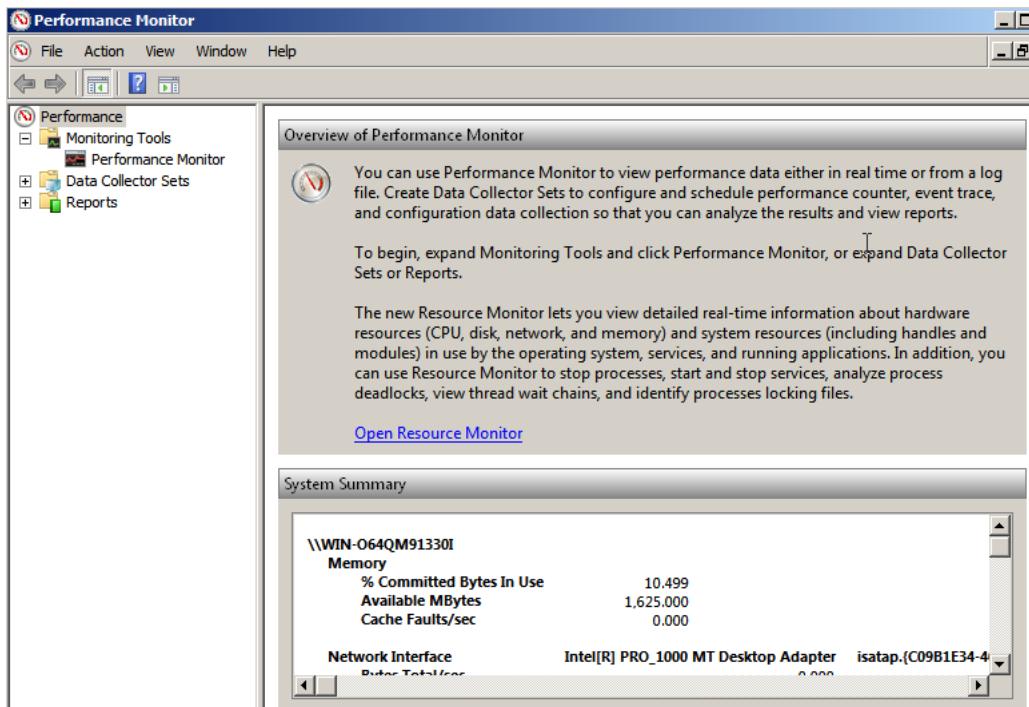


Figura 4.2: Perfmon paso 2

Ahora lo que debemos hacer es ir a *Data Collector Sets*, y buscamos *System Performance*, pulsamos botón derecho encima y le damos a *start*, ahora estará recopilando datos durante 60 s.

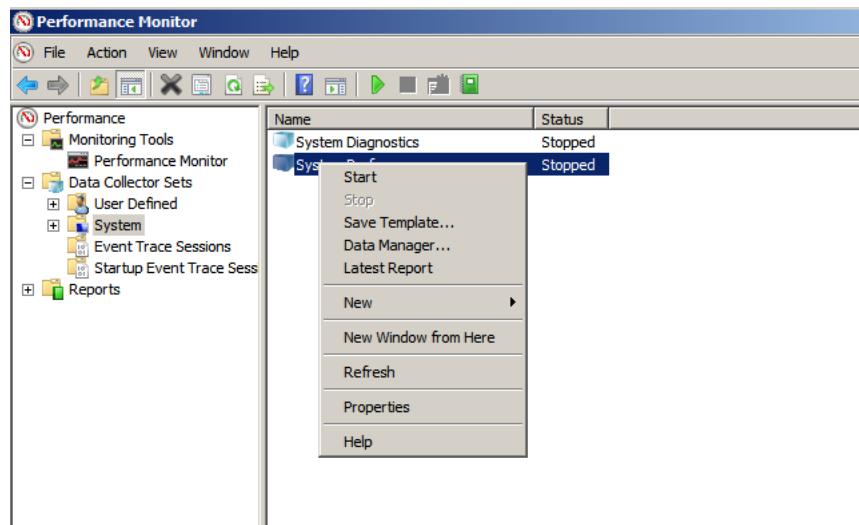


Figura 4.3: Perfmon paso 3

Si vamos a *reports*, podemos apreciar que está recopilando datos,

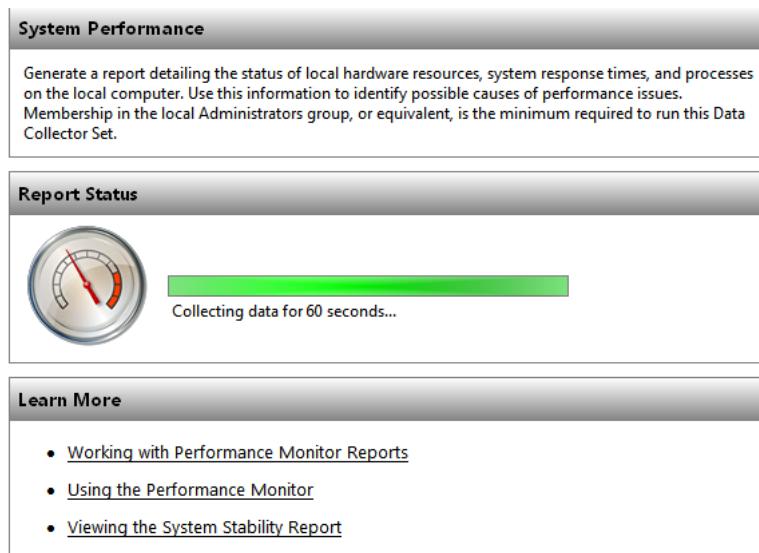


Figura 4.4: Perfmon paso 4

Una vez ha terminado nos muestra la información que ha estado recopilando, nos muestra cuando se ejecutó el recopilador, y la información relativa al equipo donde se lanzó.

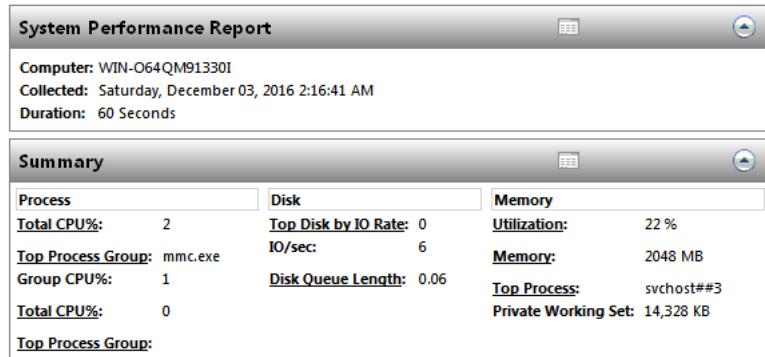


Figura 4.5: Perfmon paso 5

En esta segunda foto se puede apreciar que nos hace un resumen de la carga de la CPU, red, discos y memoria ram.

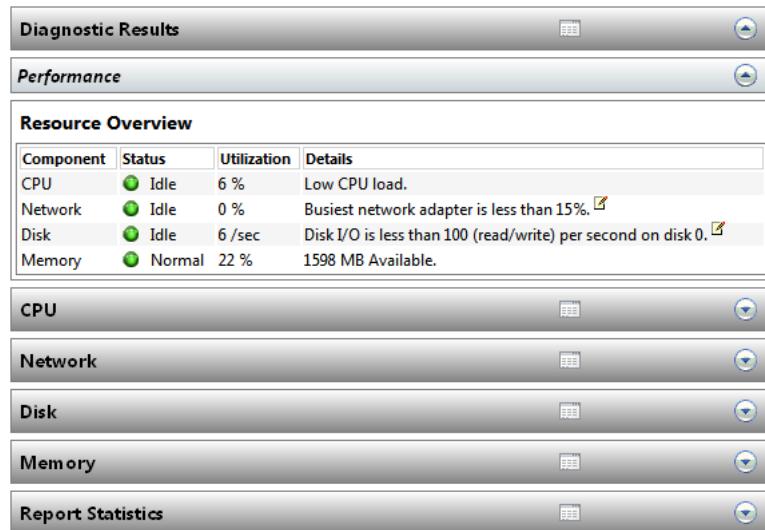


Figura 4.6: Perfmon paso 6

Toda esta información se puede ver más detallada en las pestañas de abajo, tales como procesos que se han ejecutado, qué servicios consumían más recursos, qué servicio usó la red y un largo etcétera

5. Cree un recopilador de datos definido por el usuario (modo avanzado) que incluya tanto el contador de rendimiento como los datos de seguimiento: Todos los referentes al procesador, al proceso y al servicio web, Intervalo de muestra 15 segundos, Almacene el resultado en el directorio Escritorio/logs, Incluya las capturas de pantalla de cada paso.

Lo primero que debemos hacer es dar botón derecho sobre *user defined*, y le damos a new Data Collector Sets

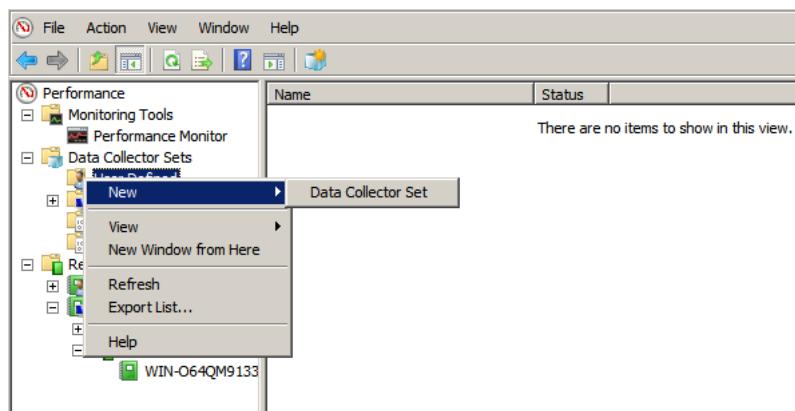


Figura 5.1: Recopilador paso 1

Seleccionamos el nombre que le queremos poner y la opción de modo avanzado.

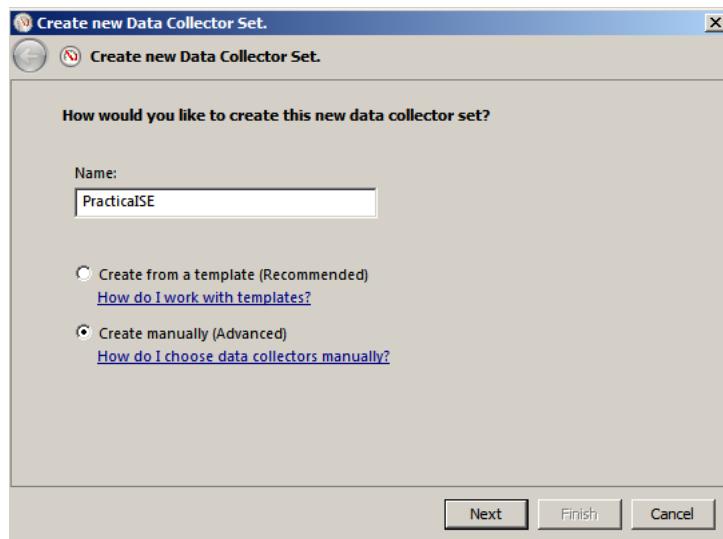


Figura 5.2: Recopilador paso 2

Ahora debemos seleccionar *Performance Counter* y *Events trace data* y le damos a siguiente.

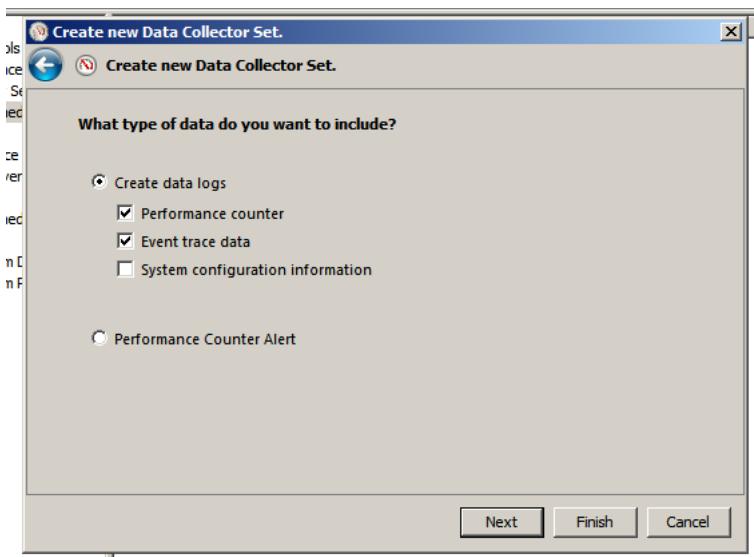


Figura 5.3: Recopilador paso 3

En este paso debemos seleccionar que queremos que nos monitorice, yo he seleccionado *Process*, *Processor*, *Processor Information* e *IIS Global*, para que monitorice el procesador, los procesos e IIS.

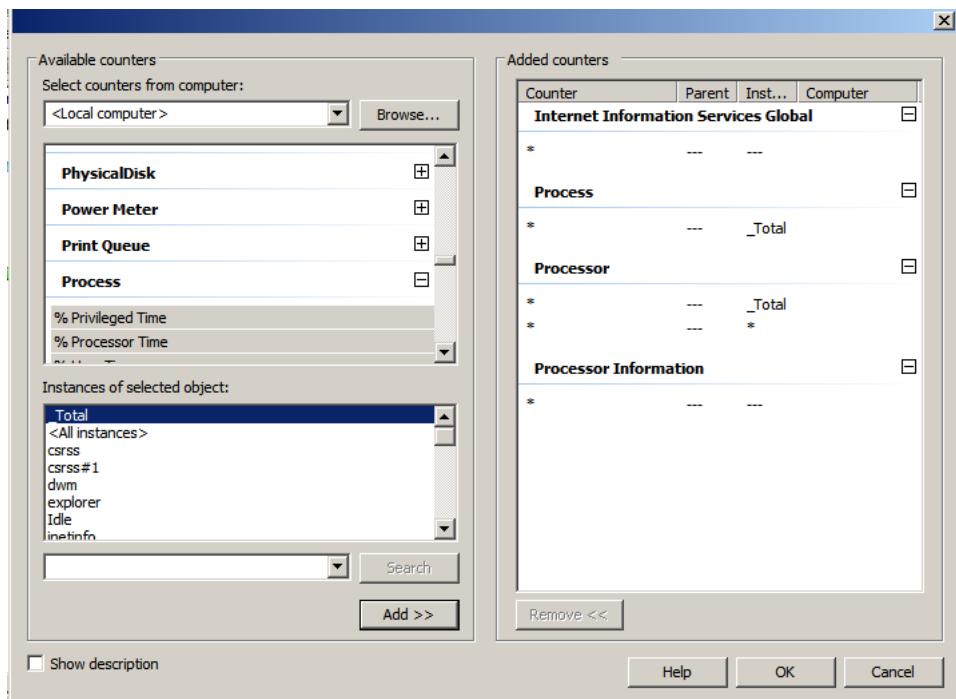


Figura 5.4: Recopilador paso 4

Seleccionamos cada cuánto tiempo queremos que haga un muestreo, en nuestro caso 15 segundos y damos a next.

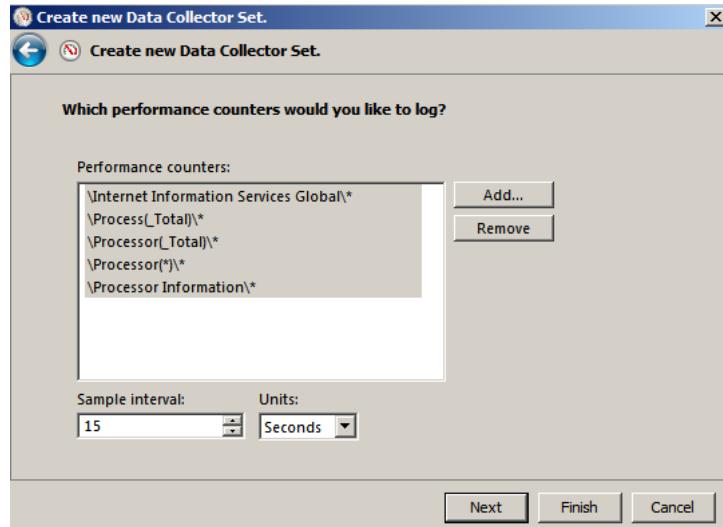


Figura 5.5: Recopilador paso 5

Debemos seleccionar los eventos que queremos muestrear, ahora seleccionaremos los que

tengan que ver con IIS.

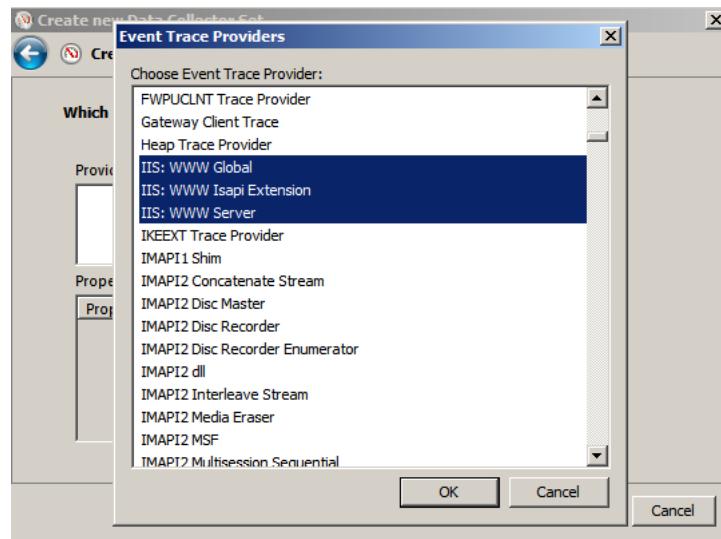


Figura 5.6: Recopilador paso 6

Definimos donde queremos que se guarden los datos del recopilador: *Escritorio/Logs*

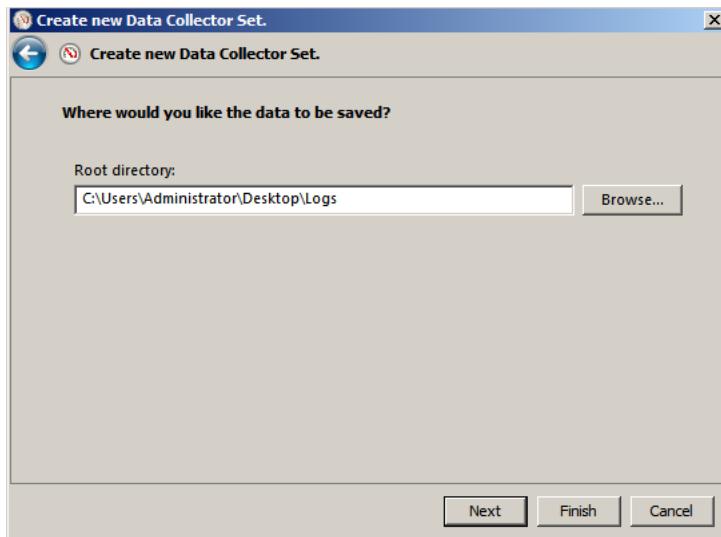


Figura 5.7: Recopilador paso 7

Y finalmente damos en finish.

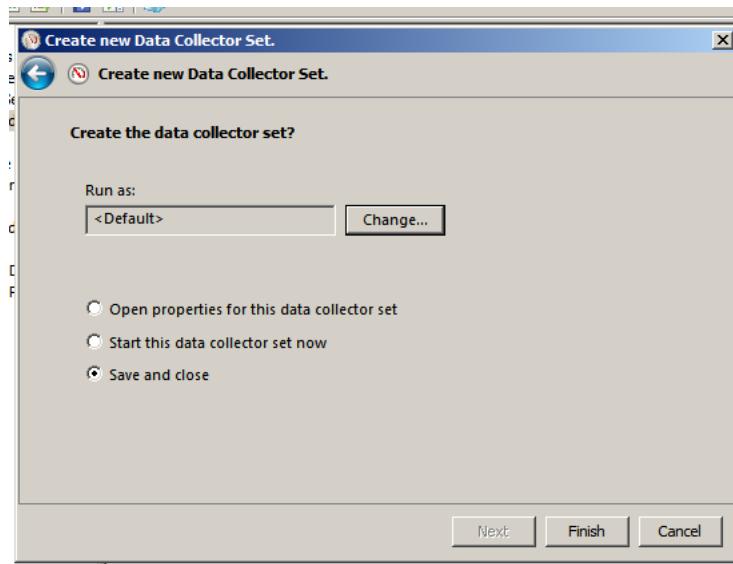


Figura 5.8: Recopilador paso 8

Vamos a ejecutar nuestro recopilador, damos a *start*

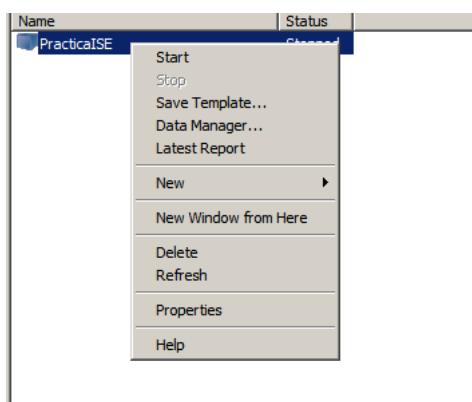


Figura 5.9: Recopilador paso 9

Ahora estará recopilando datos cada 15 segundos hasta que lo paremos.

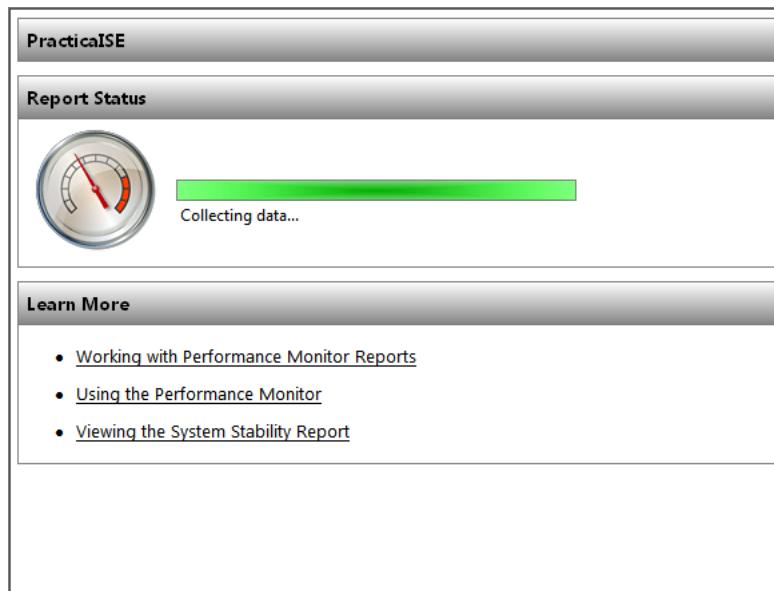


Figura 5.10: Recopilador paso 10

Si vamos a nuestro escritorio podemos observar la carpeta que nos ha generado, así como los datos que ha recopilado.

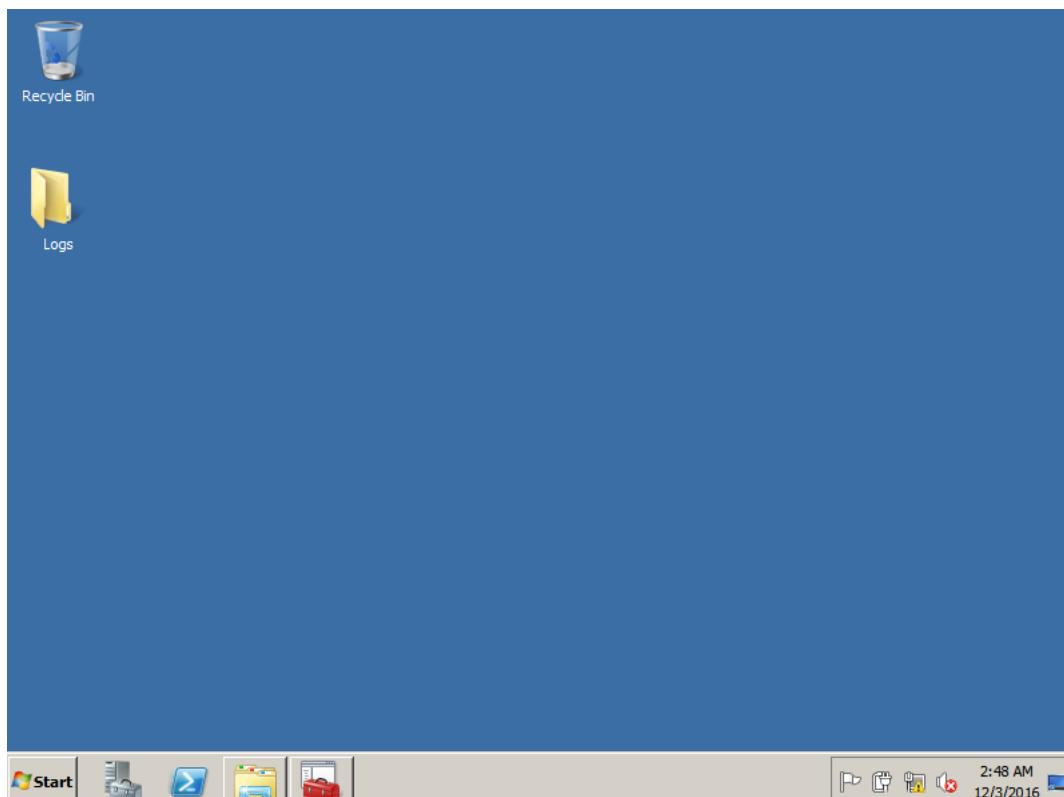


Figura 5.11: Recopilador paso 11

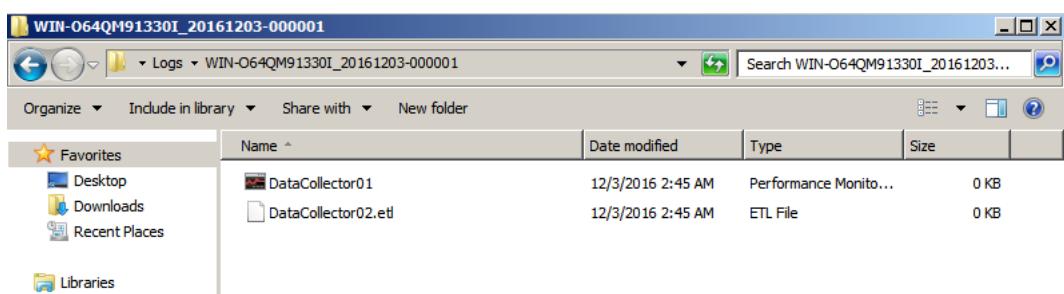


Figura 5.12: Recopilador paso 12

Vamos a inspeccionar los datos de DataCollector01, damos doble click y se nos abre.

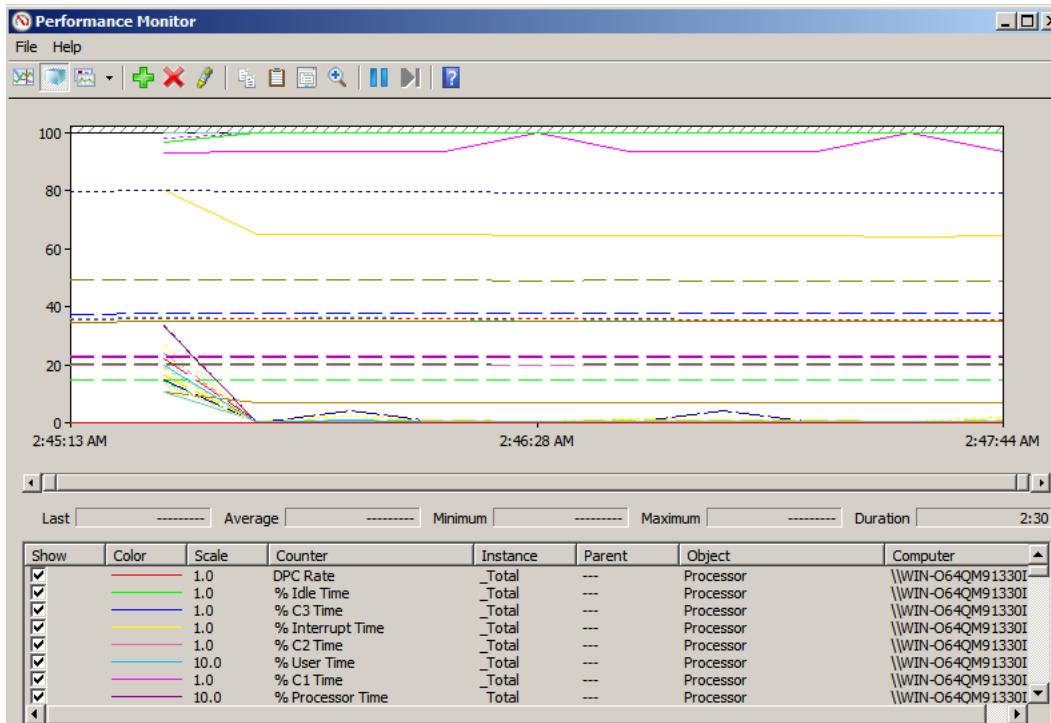


Figura 5.13: Recopilador paso 13

Ya tenemos nuestro recopilador funcionando. [1]

6. Visite la web del proyecto y acceda a la demo que proporcionan (<http://demo.munin-monitoring.org/>) donde se muestra cómo monitorizan un servidor. Monitorice varios parámetros y haga capturas de pantalla de lo que está mostrando comentando qué observa.

Accedemos a la página de la demo de Munin [5] y lo que nos encontramos nada mas entrar es

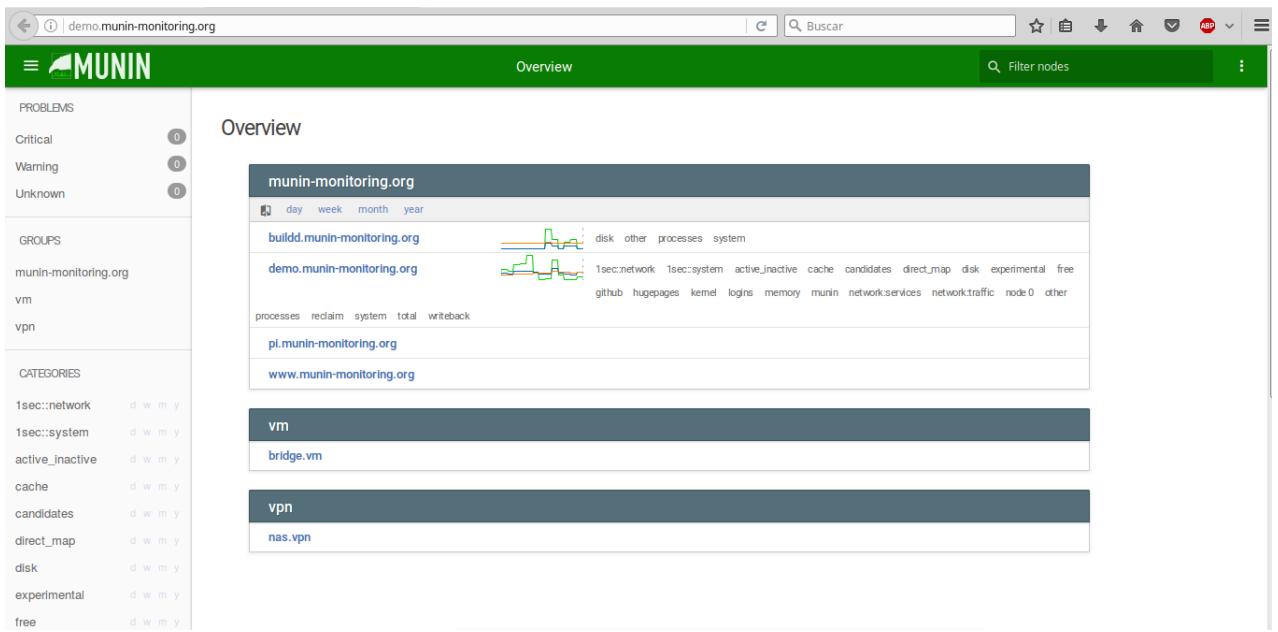


Figura 6.1: Munin 1

Seleccionamos el segundo que es lo que más está monitorizando, y vamos a visualizar *disk*



Figura 6.2: Munin 2

Pinchamos y nos muestra 4 gráficas, en una está lo que ha monitorizado durante el día, otra durante la semana,mes y año.

Como se puede apreciar en la gráfica de día, es evidente cuando realizan más operaciones

en disco, así como cuando menos se utiliza el disco, que es el sábado a las 1:00.

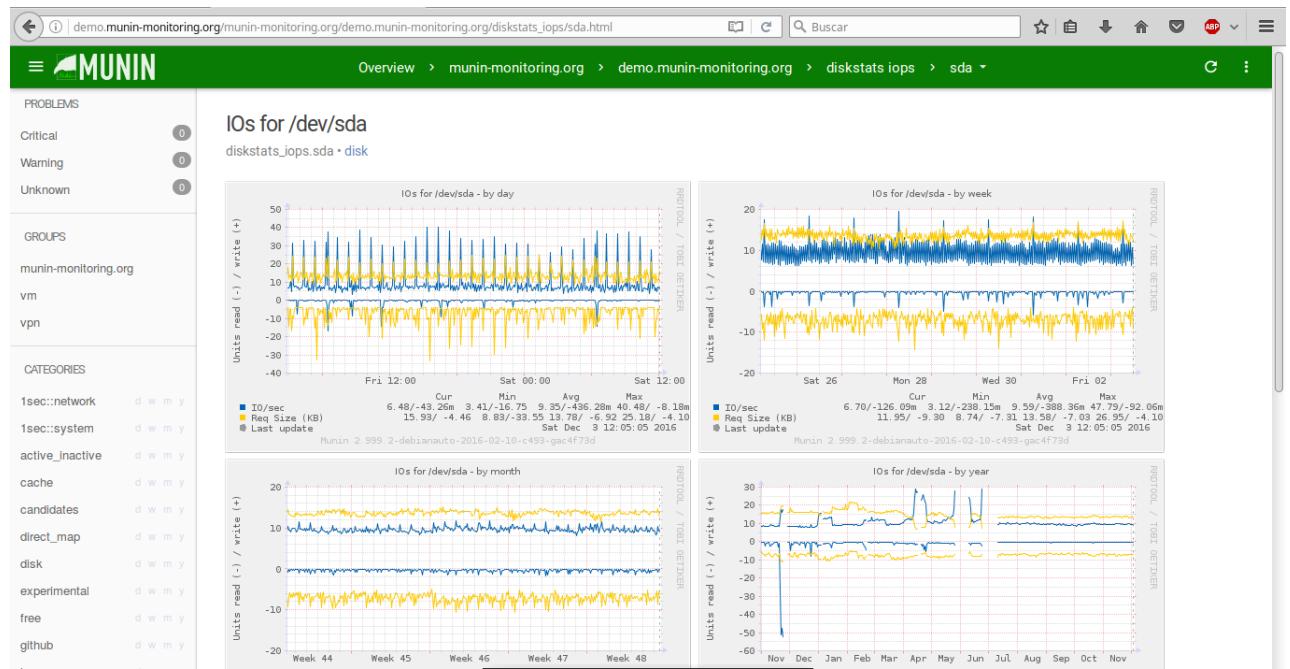


Figura 6.3: Munin 3

Al igual que con el uso de disco vamos a inspeccionar la red,

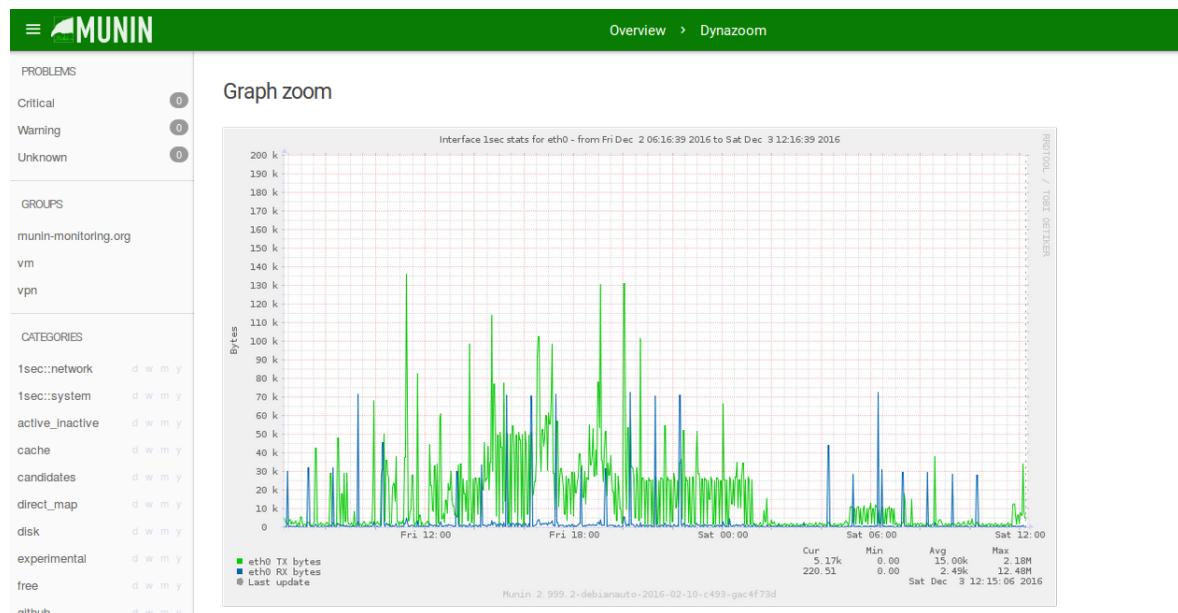


Figura 6.4: Munin 4

También al igual que los discos, se puede apreciar que cuando menos actividad de red tiene coincide con el mismo punto que los discos, lo que nos quiere decir que es cuando el sistema apenas se está utilizando, ya que la red apenas tiene tráfico.

7. Escriba un breve resumen sobre alguno de los artículos donde se muestra el uso de strace o busque otro y coméntelo.

Strace es una herramienta en línea de comando que sirve para poder visualizar las llamadas al sistema que realiza un determinado programa, así como ver cuantos threads lanza para su ejecución, podemos obtener estadísticas de el número de llamadas al sistema que ha realizado, el porcentaje de ejecución que ha consumido ese tipo de llamada y así como el numero de errores de cada una de ellas.

Strace es una buena herramienta para poder detectar que ocurre entre nuestro programa y el sistema, si algo parece que no debe ir bien.

8. Escriba un script en Python o PHP y analice su comportamiento usando el profiler presentado.

Mi script está realizado en python3 [8], genero dos matrices con números aleatorios, y luego creo una última con la multiplicación de las posiciones anteriores, $\text{sol}(i,j) = a(i,j) * b(i,j)$

```
import random

filas = 1000
columnas = 1000

matrizA= []
for i in range(filas):
    matrizA.append([])
    for j in range(columnas):
        matrizA[i].append(random.randint(0,100))

matrizB= []
for i in range(filas):
    matrizB.append([])
    for j in range(columnas):
        matrizB[i].append(random.randint(0,100))

matrizSol=[]
```

```

for i in range(filas):
    matrizSol.append([])
    for j in range(columnas):
        matrizSol[i].append((matrizA[i][j]*matrizB[i][j]))

print(matrizSol)

```

Ahora básicamente lo que haré sera ejecutarlo con el profiler de python3 mediante *python3 -m cProfile -o salida script.py*,

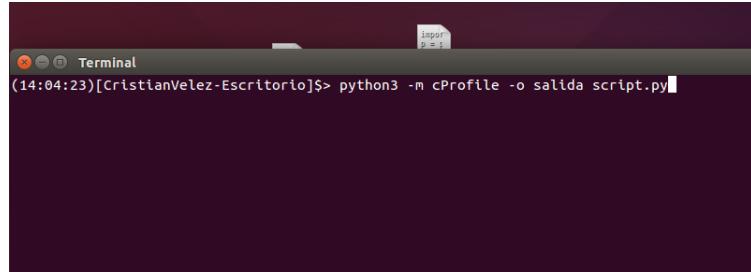


Figura 8.1: Python 1

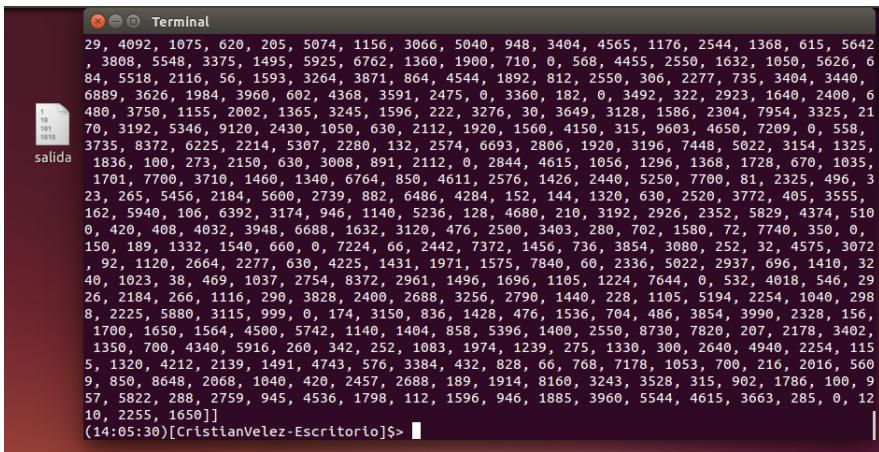


Figura 8.2: Python 2

Ahora nos ha generado un archivo que no es legible llamado salida, para poder visualizarlo crearemos otro pequeño script que nos mostrará las estadísticas legibles,

```

import pstats

p = pstats.Stats('salida')
p.strip_dirs().sort_stats(-1).print_stats()

```

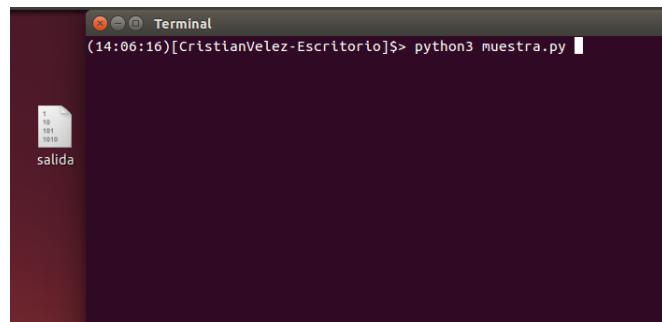


Figura 8.3: Python 3

Como se puede apreciar nos muestra todas las llamadas que ha generado el script,

```
(14:06:16)[CristianVelez-Escritorio]$> python3 muestra.py
Sun Dec  4 14:05:30 2016    salida

      13537891 function calls (13537871 primitive calls) in 18.180 seconds

Ordered by: standard name

  ncalls  tottime  percall  cumtime  percall filename:lineno(function)
      5    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1000(__init__)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1019(init_module_
ttrs)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1099(create)
  2/1    0.000    0.000    0.004    0.004 <frozen importlib._bootstrap>:1122(exec)
      3    0.000    0.000    0.002    0.001 <frozen importlib._bootstrap>:1156(_load_backwa
d_compatible)
  5/1    0.000    0.000    0.004    0.004 <frozen importlib._bootstrap>:1186(_load_unlock
d)
      5    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1266(find_spec)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1287(load_module)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:129(_new_module)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1311(is_package)
      3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1336(find_spec)
      3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:141(__init__)
      3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:144(_enter_)
  2/1    0.000    0.000    0.004    0.004 <frozen importlib._bootstrap>:1465(exec_module)
      3    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:147(__exit__)
     12    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:148(<genexpr>)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1534(get_code)
      2    0.000    0.000    0.000    0.000 <frozen importlib._bootstrap>:1591(__init__)
```

Figura 8.4: Python 4

Aquí podemos apreciar las llamadas a random que ha usado para poder generar los números aleatorios de las matrices así como inicializar la semilla una vez y luego 2.000.000 de llamadas a random y como era de esperar es el número de aleatorios que generamos, 1000x1000 = 1.000.000 cada matriz, si tenemos dos sería 2.000.000 de números aleatorios en total.

```
2000000  1.644  0.000  3.841  0.000 random.py:170(randrange)
2000000  0.900  0.000  4.741  0.000 random.py:214(randint)
2000000  1.612  0.000  2.197  0.000 random.py:220(_randbelow)
  1  0.000  0.000  0.003  0.003 random.py:37(<module>)
  1  0.000  0.000  0.000  0.000 random.py:639(SystemRandom)
  1  0.000  0.000  0.000  0.000 random.py:68(Random)
  1  0.000  0.000  0.000  0.000 random.py:84(__init__)
  1  0.000  0.000  0.000  0.000 random.py:93(seed)
```

Figura 8.5: Python 5

9. Acceda a la consola mysql (o a través de phpMyAdmin) y muestre el resultado de mostrar el "profile" de una consulta (la creación de la BD y la consulta la puede hacer libremente).

Primero lo que voy a hacer es acceder a mysql [7] desde la terminal,

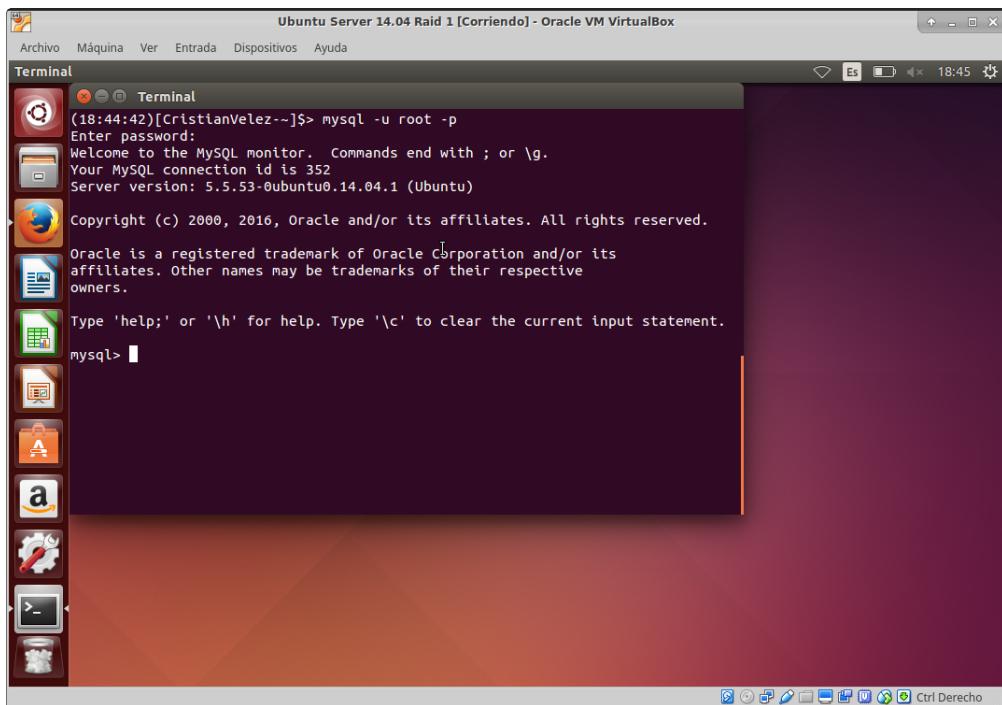


Figura 9.1: Profiler mysql 1

Una vez estamos dentro vamos a decirle a mysql que base de datos queremos utilizar para ello usaremos *USE mysql;*

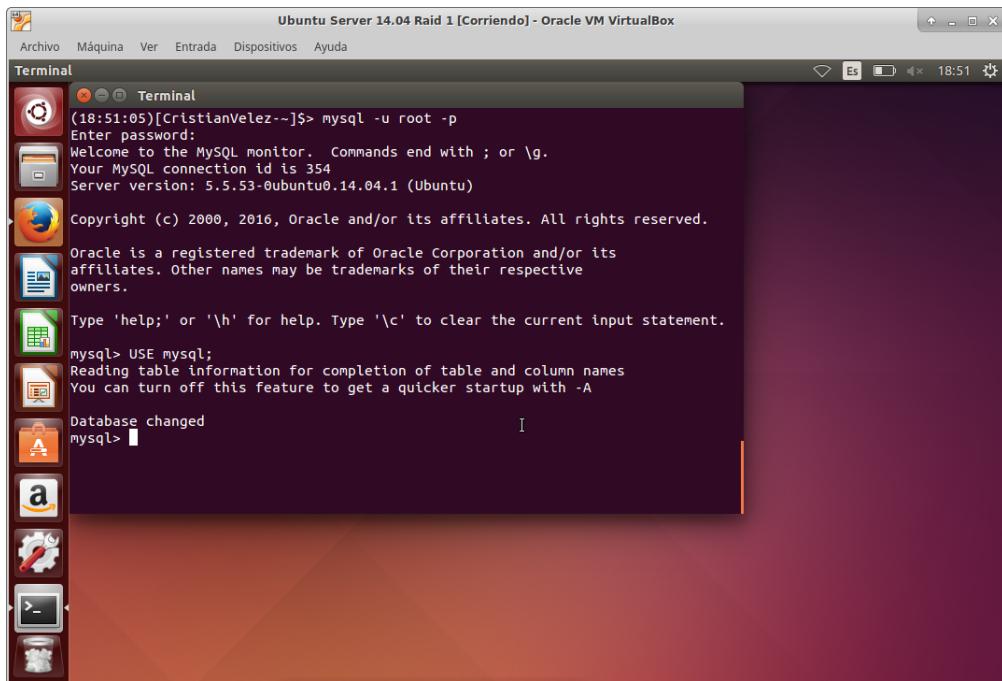


Figura 9.2: Profiler mysql 2

Ahora debemos activar el profiling , con $SET profiling=1;$; lo que hagamos a partir de ahora quedará registrado su tiempo y estadística de cada operación que realicemos.

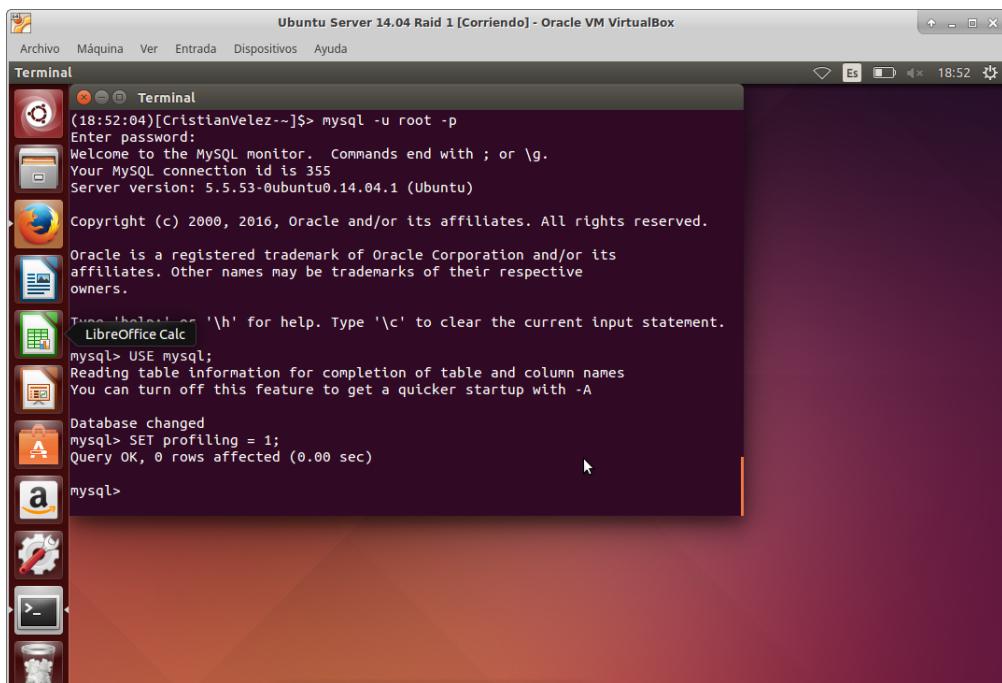


Figura 9.3: Profiler mysql 3

Intentamos crear una tabla alumnos, pero como anteriormente fue creada como prueba da error.

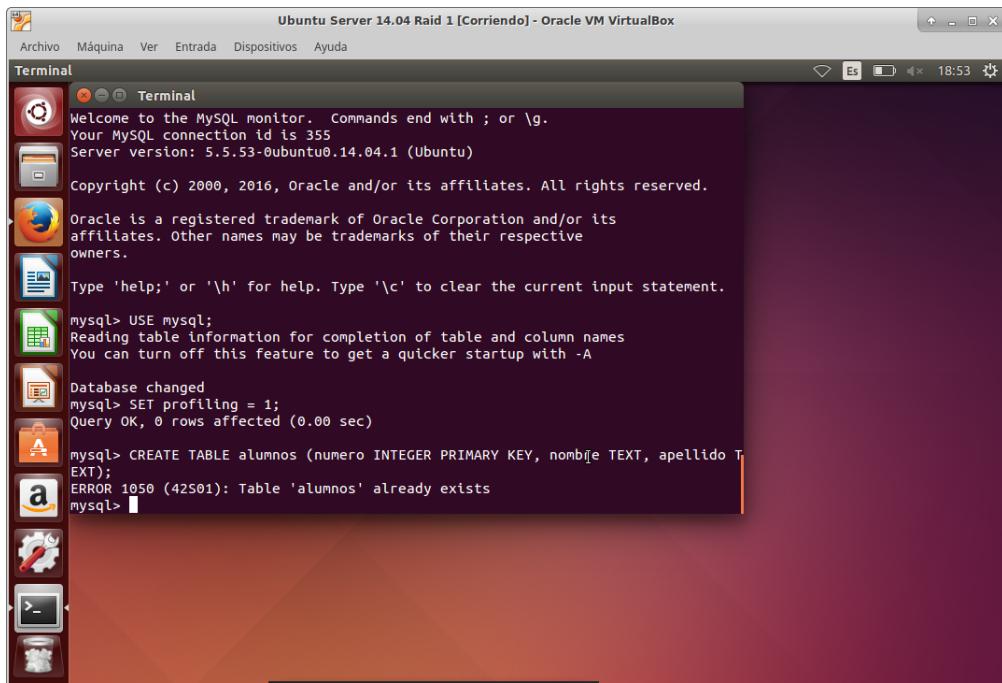


Figura 9.4: Profiler mysql 4

Eliminamos la tabla anterior con *DROP alumnos*; y creamos una nueva de la misma manera que se intentó anteriormente.

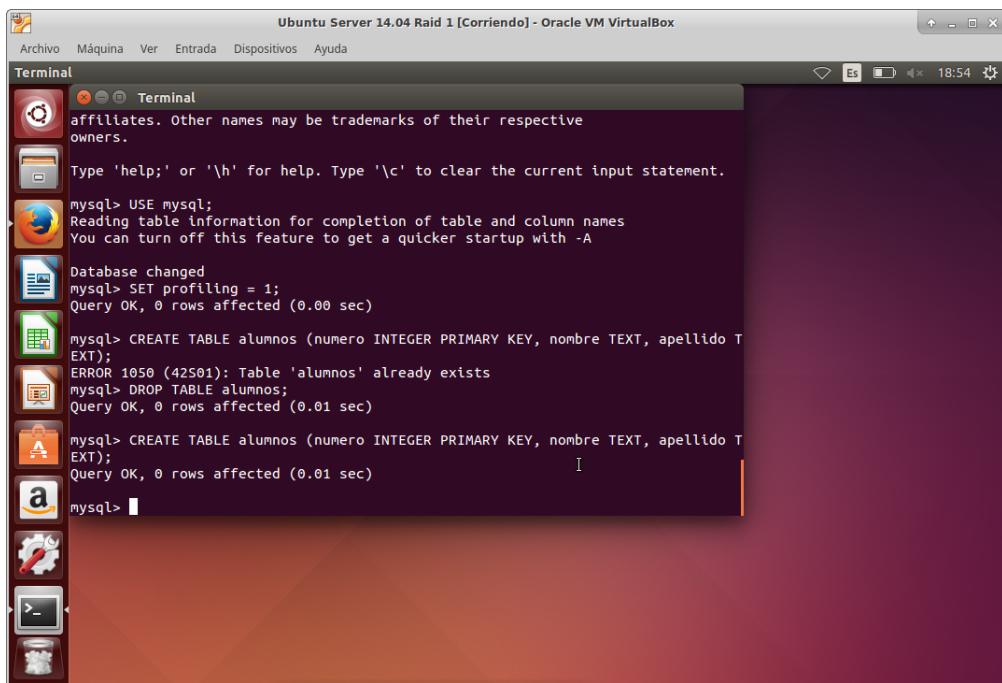


Figura 9.5: Profiler mysql 5

Ahora voy a insertar los datos de los alumnos para poder realizar una consulta.

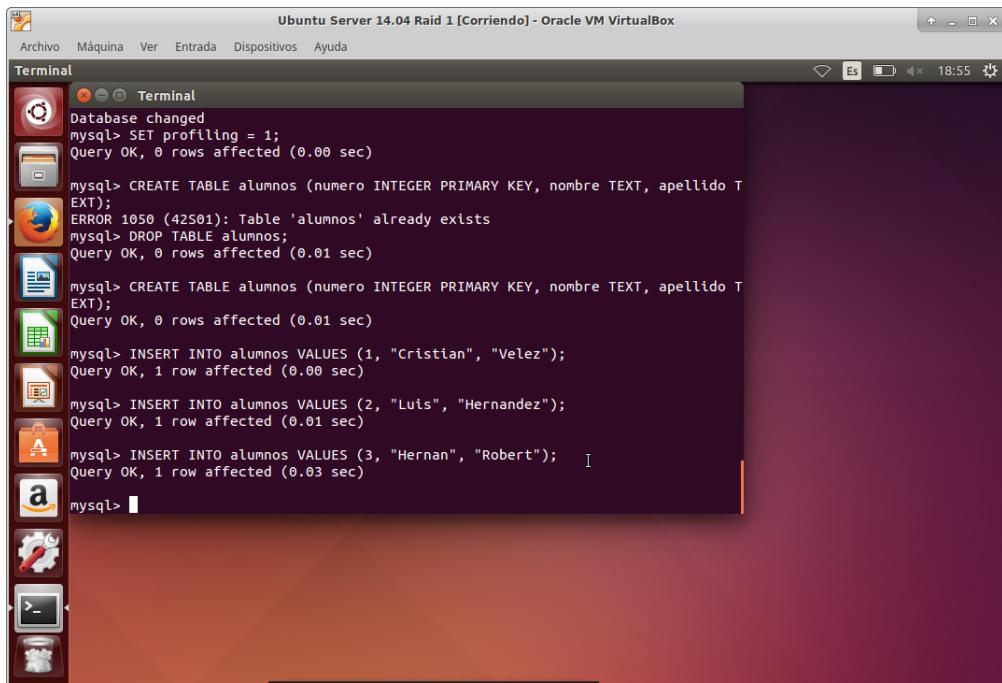


Figura 9.6: Profiler mysql 6

Una vez ya está rellena la tabla, voy a hacer una consulta para ver si obtengo como me llamo a partir de mi apellido con *SELECT nombre FROM alumnos WHERE apellido="Velez";*

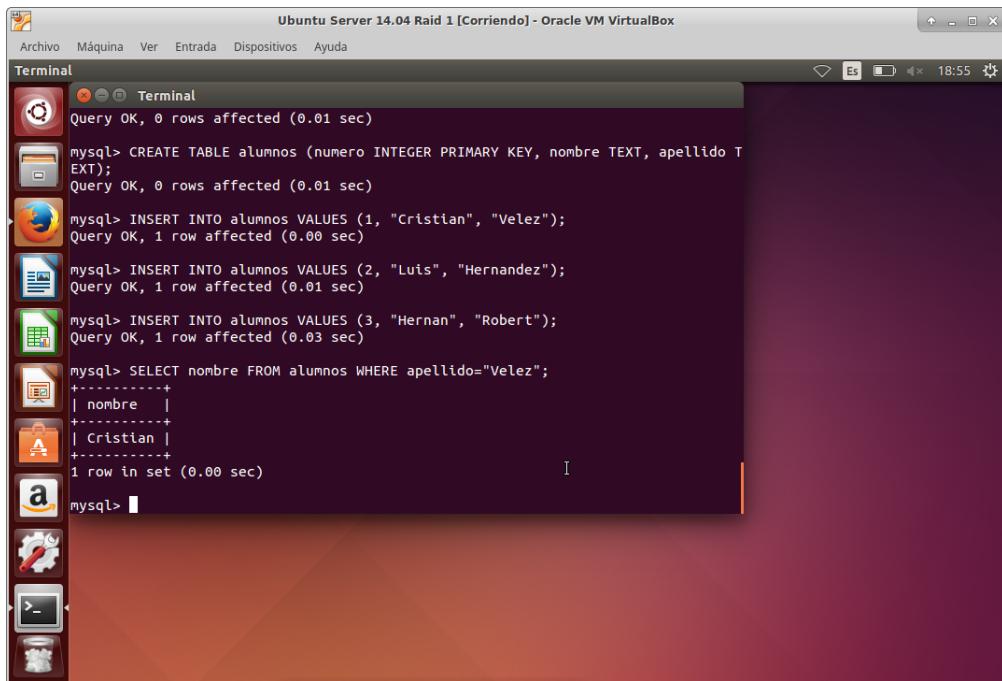


Figura 9.7: Profiler mysql 7

Una vez terminadas las operaciones vamos a ver lo que ha registrado el profiler de mysql, para ello basta con *SHOW PROFILES*;

Query_ID	Duration	Query
1	0.00023775	CREATE TABLE alumnos (numero INTEGER PRIMARY KEY, nombre TEXT, apellido TEXT)
2	0.00825700	DROP TABLE alumnos
3	0.01072750	CREATE TABLE alumnos (numero INTEGER PRIMARY KEY, nombre TEXT, apellido TEXT)
4	0.00695250	INSERT INTO alumnos VALUES (1, "Cristian", "Velez")
5	0.00745925	INSERT INTO alumnos VALUES (2, "Luis", "Hernandez")
6	0.03278550	INSERT INTO alumnos VALUES (3, "Hernan", "Robert")
7	0.00030725	SELECT nombre FROM alumnos WHERE apellido="Velez"

Figura 9.8: Profiler mysql 8

Ya podemos ver el tiempo que nos ha tardado en cada una de las operaciones y ver cuales pueden ser más costosas para nuestro sistema, en mi caso ha sido al crear la tabla

alumnos.

Referencias

- [1] Creación de recopilador de datos. [https://technet.microsoft.com/es-es/library/cc722148\(v=ws.11\).aspx](https://technet.microsoft.com/es-es/library/cc722148(v=ws.11).aspx), Diciembre de 2016.
- [2] Documentación de cron. <https://wiki.gentoo.org/wiki/Cron/es>, Diciembre de 2016.
- [3] Documentación de crontab. <https://linux.die.net/man/5/crontab>, Diciembre de 2016.
- [4] Manual de dpkg. <http://man7.org/linux/man-pages/man1/dpkg.1.html>, Diciembre de 2016.
- [5] Página de demos de munin. <http://demo.munin-monitoring.org/>, Diciembre de 2016.
- [6] Uso de perfmon. [https://technet.microsoft.com/es-es/library/cc749115\(v=ws.11\).aspx](https://technet.microsoft.com/es-es/library/cc749115(v=ws.11).aspx), Diciembre de 2016.
- [7] Uso de profiler para mysql. <https://dev.mysql.com/doc/refman/5.5/en/show-profile.html>, Diciembre de 2016.
- [8] Uso de profiler para python 3. <https://docs.python.org/3/library/profile.html>, Diciembre de 2016.