

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

Memoria Práctica 3

Guillermo Montes Martos

9 de diciembre de 2016

Índice

1. Cuestión 1	4
1.1. a) ¿Qué archivo le permite ver qué programas se han instalado con el gestor de paquetes?	4
1.2. b) ¿Qué significan las terminaciones .1.gz o .2.gz de los archivos en ese directorio?	4
2. Cuestión 2	5
2.1. ¿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).	5
3. Cuestión 3	6
3.1. Pruebe a ejecutar el comando dmesg, 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. . . .	6
4. Cuestión 4	7
4.1. Ejecute el monitor de “System Performance” y muestre el resultado. Incluya capturas de pantalla comentando la información que aparece.	7
5. Cuestión 5	9
5.1. Cree un recopilador de datos definido por el usuario (modo avanzado) con perfmon 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. . .	9
6. Cuestión 6	14
6.1. Visite la web del proyecto y acceda a la demo que proporcionan (http://demo.munin-monitoring.org/) donde se muestra cómo monitorizan un servidor. Monitoree varios parámetros y haga capturas de pantalla de lo que está mostrando comentando qué observa.	14
7. Cuestión 7	15
7.1. Escriba un breve resumen sobre alguno de los artículos donde se muestra el uso de strace o busque otro y coméntelo.	15
8. Cuestión 8	16
8.1. Escriba un script en Python o PHP y analice su comportamiento usando el profiler presentado.	16

9. Cuestión 9	18
9.1. 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).	18

Índice de figuras

1.1. Últimos 20 paquetes instalados registrados por <i>dpkg</i> .	4
2.1. Archivo <i>/etc/crontab</i> donde se ha programado el trabajo.	5
3.1. Salida del comando <i>dmesg</i> antes de insertar una memoria USB.	6
3.2. Salida del comando <i>dmesg</i> después de insertar una memoria USB.	7
4.1. Ejecución del monitor "System Performance" en Windows Server.	8
4.2. Resultados generales del monitor "System Performance" en Windows Server.	8
4.3. Resultado de memoria del monitor "System Performance" en Windows Server.	9
5.1. Creando nuevo conjunto de recopiladores de datos.	10
5.2. Interfaz para la creación de conjuntos de recopiladores de datos.	10
5.3. Selección de tipo de datos.	11
5.4. Selección de contadores de rendimiento a registrar.	11
5.5. Confirmación de contadores de rendimiento a registrar.	12
5.6. Selección de ubicación de guardado.	12
5.7. Paso final para la creación del conjunto de recopiladores de datos.	13
5.8. Resultado de memoria del monitor "System Performance" en Windows Server.	13
6.1. Latencia de disco por dispositivo por día y semana.	14
6.2. Número de hilos en ejecución por día y semana.	15
8.1. Resultado de la ejecución del script <i>profiler</i> en <i>python</i> .	18
9.1. Activación del <i>profiler</i> en phpMyAdmin.	19
9.2. Consulta sobre la tabla creada con phpMyAdmin.	19
9.3. Datos obtenidos por el <i>profiler</i> sobre la consulta realizada con phpMyAdmin.	20
9.4. Resultado de la consulta realizada con phpMyAdmin.	20

1. Cuestión 1

1.1. a) ¿Qué archivo le permite ver qué programas se han instalado con el gestor de paquetes?

La pregunta como tal es ambigua, ya que no especifica el gestor de paquetes al que hace referencia. Por lo tanto, se ha decidido responder acerca de Ubuntu Server, donde además existen dos gestores de paquetes predeterminados cuya principal diferencia es la posibilidad de descargarlos, *dpkg* [1] y *apt* [2].

Si nos referimos al primero de los dos, el archivo requerido sería `/var/log/dpkg.log`, donde filtrando mediante la cadena `"installed"`, obtendríamos una lista con los paquetes instalados. Por otro lado, para *apt* deberíamos de consultar el fichero `/var/log/apt/history.log`, donde podemos obtener una lista con los paquetes instalados filtrando mediante la cadena `"Install:"`.

```
[07/12/16 gmm@ubuntuserver:~] $ cat /var/log/dpkg.log | grep installed | tail -n 20
2016-12-04 18:49:10 status installed linux-headers-4.4.0-51:all 4.4.0-51.72
2016-12-04 18:49:10 status installed linux-headers-4.4.0-51-generic:amd64 4.4.0-51.72
2016-12-04 18:49:10 status installed linux-headers-generic:amd64 4.4.0.51.54
2016-12-04 18:49:10 status installed linux-generic:amd64 4.4.0.51.54
2016-12-04 18:49:10 status installed linux-libc-dev:amd64 4.4.0-51.72
2016-12-04 18:49:11 status installed liblxc1:amd64 2.0.5-0ubuntu1~ubuntu16.04.3
2016-12-04 18:49:14 status installed lxc-common:amd64 2.0.5-0ubuntu1~ubuntu16.04.3
2016-12-04 18:49:14 status installed libc-bin:amd64 2.23-0ubuntu3
2016-12-07 18:30:57 status half-installed linux-image-4.4.0-53-generic:amd64 4.4.0-53.74
2016-12-07 18:31:00 status half-installed linux-image-extra-4.4.0-53-generic:amd64 4.4.0-53.74
2016-12-07 18:31:09 status half-installed linux-generic:amd64 4.4.0.51.54
2016-12-07 18:31:09 status half-installed linux-generic:amd64 4.4.0.51.54
2016-12-07 18:31:09 status half-installed linux-image-generic:amd64 4.4.0.51.54
2016-12-07 18:31:09 status half-installed linux-image-generic:amd64 4.4.0.51.54
2016-12-07 18:31:09 status half-installed linux-headers-4.4.0-53:all 4.4.0-53.74
2016-12-07 18:31:21 status half-installed linux-headers-4.4.0-53-generic:amd64 4.4.0-53.74
2016-12-07 18:31:25 status half-installed linux-headers-generic:amd64 4.4.0.51.54
2016-12-07 18:31:25 status half-installed linux-headers-generic:amd64 4.4.0.51.54
2016-12-07 18:31:25 status half-installed linux-libc-dev:amd64 4.4.0-51.72
2016-12-07 18:31:25 status half-installed linux-libc-dev:amd64 4.4.0-51.72
[07/12/16 gmm@ubuntuserver:~] $ _
```

Figura 1.1: Últimos 20 paquetes instalados registrados por *dpkg*.

1.2. b) ¿Qué significan las terminaciones `.1.gz` o `.2.gz` de los archivos en ese directorio?

No se ha encontrado una referencia válida para esta cuestión, por lo cual se responderá en base a un tema de un foro de Ubuntu [3].

Tal y como se explica en este, son antiguos archivos de registro, los cuales podemos encontrar tanto para *dpkg* como para *apt*, que son rotados cada cierto tiempo y comprimidos, de manera que podamos conservar un registro de la actividad anterior del gestor de paquetes. Para poder visualizar su contenido, tendríamos que descomprimir los ficheros o visualizarlos directamente con el comando *zless* [4].

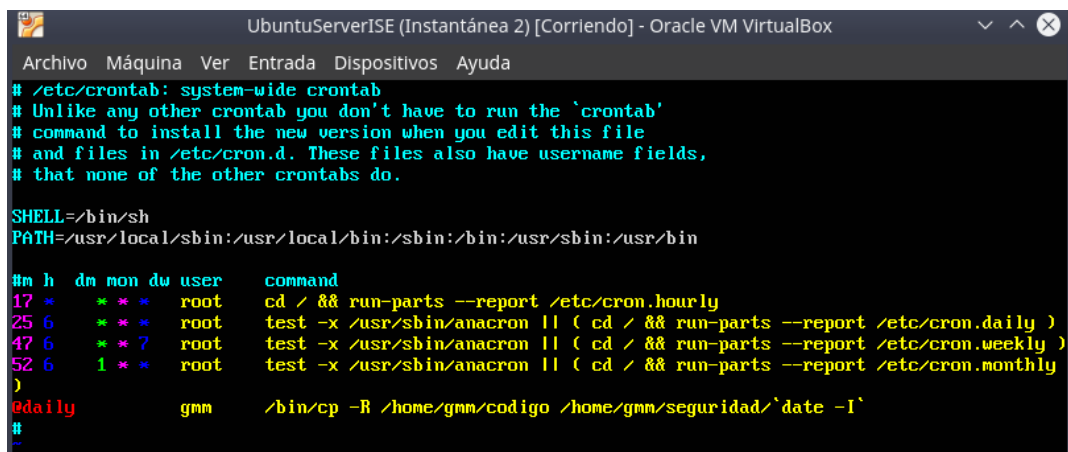
2. Cuestión 2

2.1. ¿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 este ejercicio, consultamos el manual de cron [5]. Este nos explica que cron es la herramienta, el demonio que ejecuta las tareas programadas, mientras que crontab [6] es el nombre que reciben los archivos que maneja el demonio cron, es decir, donde se encuentran dichas tareas programadas. Es en la documentación de este último es donde encontramos el archivo que debemos modificar para programar una tarea, el cual se trata de `/etc/crontab`. También podemos añadir otros trabajos creando sendos archivos en el directorio `/etc/cron.d/`.

Para crear una copia tal y como se especifica en el enunciado, lo más sencillo será usar el comando `date` [7], el cual, utilizando el argumento `-I`, nos muestra la fecha en un formato reducido. Para especificar la periodicidad de la tarea, se ha elegido usar una de las palabras reservadas de cron: `@daily`, la cual especifica que todos los días a medianoche se realizará el trabajo. Se podría haber usado el formato típico de crontab, especificando minuto, hora, día del mes, mes y día de la semana, pero ya que no se requiere que el trabajo se realice a una hora concreta, se ha elegido la primera opción por simplicidad. Como se aconseja en el manual de crontab, se usan paths absolutos. Por último, ya que los directorios se encuentran en el *home* del usuario, establecemos dicho usuario como el ejecutor del trabajo. Así, la línea quedaría de la siguiente forma teniendo en cuenta que el usuario de nuestro sistema es *gmm*:

```
@daily gmm /bin/cp -R /home/gmm/codigo "/home/gmm/seguridad/$(date -I)"
```



```
UbuntuServerISE (Instantánea 2) [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

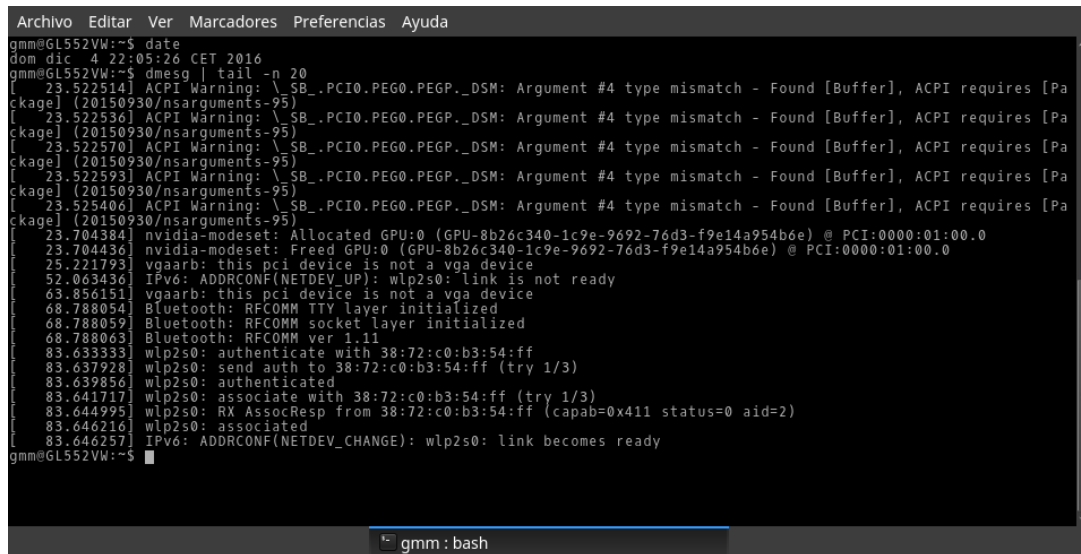
#m h  dm mon dw user    command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * ? root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
)
@daily gmm /bin/cp -R /home/gmm/codigo /home/gmm/seguridad/`date -I`
#
```

Figura 2.1: Archivo `/etc/crontab` donde se ha programado el trabajo.

3. Cuestión 3

3.1. Pruebe a ejecutar el comando `dmesg`, 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.

Tal y como se explica en el guión, el comando `dmesg` [8] se usa para examinar el buffer circular del kernel. En los ejemplos mostrados ejecutados en la máquina anfitriona, se muestran las 20 últimas líneas de la salida.



```
Archivo Editar Ver Marcadores Preferencias Ayuda
gmm@GL552VW:~$ date
dom dic  4 22:05:26 CET 2016
gmm@GL552VW:~$ dmesg | tail -n 20
[  23.522514] ACPI Warning: \_SB_.PCI0.PEG0.PEGP._DSM: Argument #4 type mismatch - Found [Buffer], ACPI requires [Package] (20150930/nsarguments-95)
[  23.522536] ACPI Warning: \_SB_.PCI0.PEG0.PEGP._DSM: Argument #4 type mismatch - Found [Buffer], ACPI requires [Package] (20150930/nsarguments-95)
[  23.522570] ACPI Warning: \_SB_.PCI0.PEG0.PEGP._DSM: Argument #4 type mismatch - Found [Buffer], ACPI requires [Package] (20150930/nsarguments-95)
[  23.522593] ACPI Warning: \_SB_.PCI0.PEG0.PEGP._DSM: Argument #4 type mismatch - Found [Buffer], ACPI requires [Package] (20150930/nsarguments-95)
[  23.525406] ACPI Warning: \_SB_.PCI0.PEG0.PEGP._DSM: Argument #4 type mismatch - Found [Buffer], ACPI requires [Package] (20150930/nsarguments-95)
23.704384] nvidia-modeset: Allocated GPU:0 (GPU-8b26c340-1c9e-9692-76d3-f9e14a954b6e) @ PCI:0000:01:00.0
23.704436] nvidia-modeset: Freed GPU:0 (GPU-8b26c340-1c9e-9692-76d3-f9e14a954b6e) @ PCI:0000:01:00.0
25.221793] vgaarb: this pci device is not a vga device
52.063436] IPv6: ADDRCONF(NETDEV_UP): wlp2s0: link is not ready
63.856151] vgaarb: this pci device is not a vga device
68.788054] Bluetooth: RFCOMM TTY layer initialized
68.788059] Bluetooth: RFCOMM socket layer initialized
68.788063] Bluetooth: RFCOMM ver 1.11
83.633333] wlp2s0: authenticate with 38:72:c0:b3:54:ff
83.637928] wlp2s0: send auth to 38:72:c0:b3:54:ff (try 1/3)
83.639856] wlp2s0: authenticated
83.641717] wlp2s0: associate with 38:72:c0:b3:54:ff (try 1/3)
83.644995] wlp2s0: RX AssocResp from 38:72:c0:b3:54:ff (capab=0x411 status=0 aid=2)
83.646216] wlp2s0: associated
83.646257] IPv6: ADDRCONF(NETDEV_CHANGE): wlp2s0: link becomes ready
gmm@GL552VW:~$
```

Figura 3.1: Salida del comando `dmesg` antes de insertar una memoria USB.

Antes de insertar la memoria USB, podemos ver mensajes acerca del WiFi, bluetooth, drivers de gráficos, etc.

```
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda
gmm@GL552VW:~$ date
dom dic  4 22:05:55 CET 2016
gmm@GL552VW:~$ dmesg | tail -n 20
83.646216] wlp2s0: associated
406.217799] IPv6: ADDRCONF(NETDEV_CHANGE): wlp2s0: link becomes ready
406.346481] usb 1-6: new high-speed USB device number 6 using xhci_hcd
406.346481] usb 1-6: New USB device found, idVendor=0781, idProduct=5590
406.346488] usb 1-6: New USB device strings: Mfr=1, Product=2, SerialNumber=3
406.346493] usb 1-6: Product: Ultra
406.346496] usb 1-6: Manufacturer: SanDisk
406.346500] usb 1-6: SerialNumber: 4C530001090416112265
406.436222] usb-storage 1-6:1.0: USB Mass Storage device detected
406.436455] scsi host3: usb-storage 1-6:1.0
406.436655] usbcore: registered new interface driver usb-storage
406.453190] usbcore: registered new interface driver uas
407.435015] scsi 3:0:0:0: Direct-Access    SanDisk  Ultra        1.00 PQ: 0 ANSI: 6
407.435690] sd 3:0:0:0: Attached scsi generic sg2 type 0
407.435802] sd 3:0:0:0: [sdb] 60062500 512-byte logical blocks: (30.8 GB/28.6 GiB)
407.436487] sd 3:0:0:0: [sdb] Write Protect is off
407.436498] sd 3:0:0:0: [sdb] Mode Sense: 43 00 00 00
407.436809] sd 3:0:0:0: [sdb] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
407.445161] sdb: sdb1
407.446517] sd 3:0:0:0: [sdb] Attached SCSI removable disk
gmm@GL552VW:~$
```

Figura 3.2: Salida del comando dmesg después de insertar una memoria USB.

Tal y como podemos observar en la captura, la máquina detecta la memoria USB, para la cual se dispone a usar el driver *xhci_hcd*. También podemos ver el identificador y el nombre del fabricante y del producto, así como el número de serie de este último. A continuación la detecta como una unidad USB de almacenamiento masivo y crea las interfaces necesarias. Monta el dispositivo sin protección de escritura y con la caché de lectura activada en el punto de montaje `/dev/sdb`. Además, se muestra el tamaño del dispositivo y la interfaz SCSI 3.0 (Small Computer System Interface) [9] que se ha creado para la transferencia de datos.

4. Cuestión 4

4.1. Ejecute el monitor de "System Performance" y muestre el resultado. Incluya capturas de pantalla comentando la información que aparece.

Para abrir el monitor, tendremos que abrir un terminal tal y como se explica en el guión de prácticas y ejecutar *perfmon* [10]. Hecho esto, nos vamos al menú de "Conjunto de recopiladores de datos" y submenú "Sistema", donde tendremos que hacer click derecho en la opción "System Performance" y darle a iniciar. Esperamos el minuto que tarda en ejecutarse y vemos los resultados haciendo click derecho en la misma opción y pinchando en "Informe más reciente".

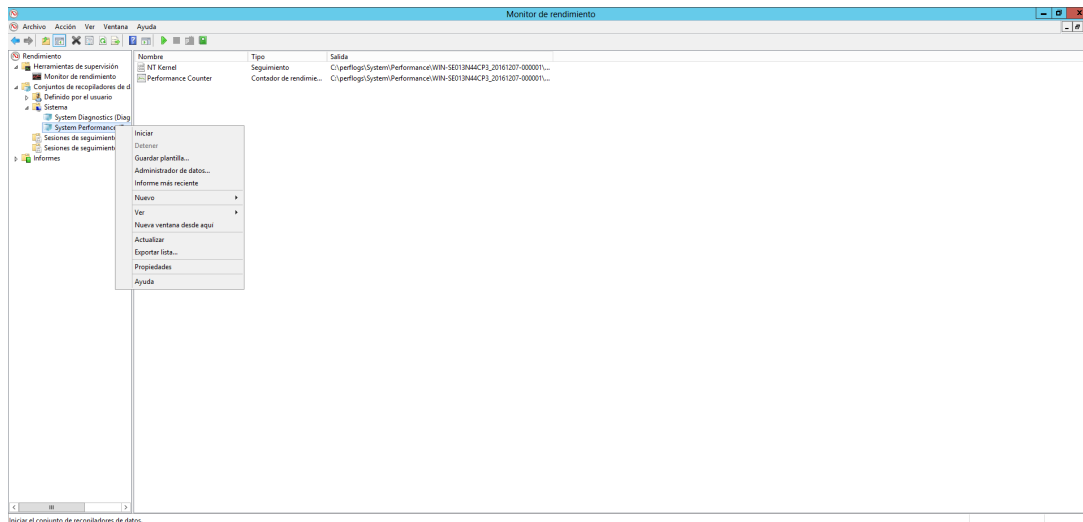


Figura 4.1: Ejecución del monitor "System Performance" en Windows Server.

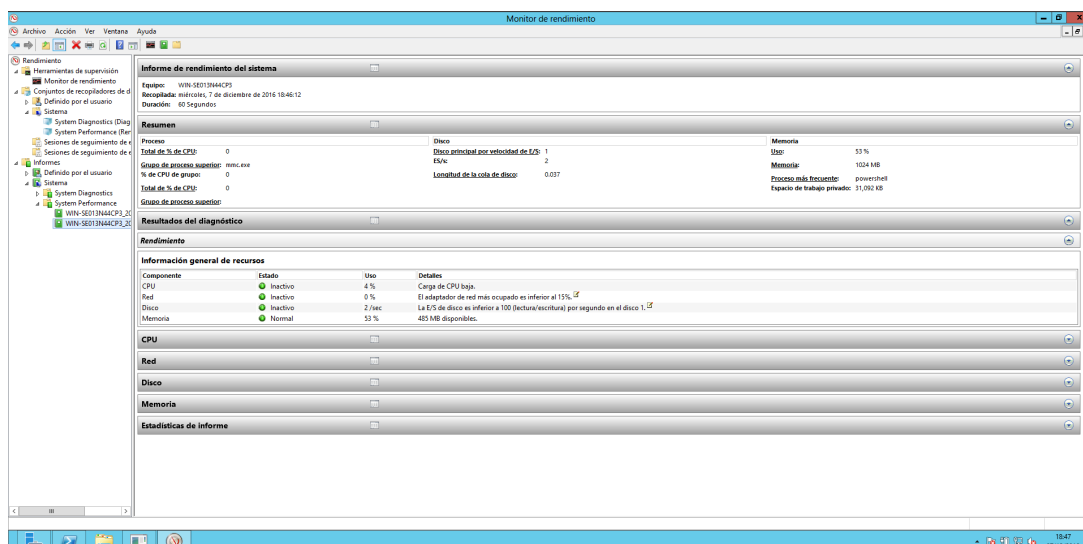


Figura 4.2: Resultados generales del monitor "System Performance" en Windows Server.

En los resultados generales del monitor, podemos comprobar que la carga del sistema es muy baja ya que no tiene apenas servicios en ejecución más allá de los predeterminados del sistema operativo. Así, los valores de carga de CPU, de red y de E/S en disco son insignificantes. Lo único que tiene un valor relativamente medio es el uso de memoria principal, la cual se encuentra a un 53 % (unos 540MB usados), un dato parcialmente normal si tenemos en cuenta los requisitos mínimos de memoria para la ejecución de Windows Server 2012, el cual se trata de 512MB [11].

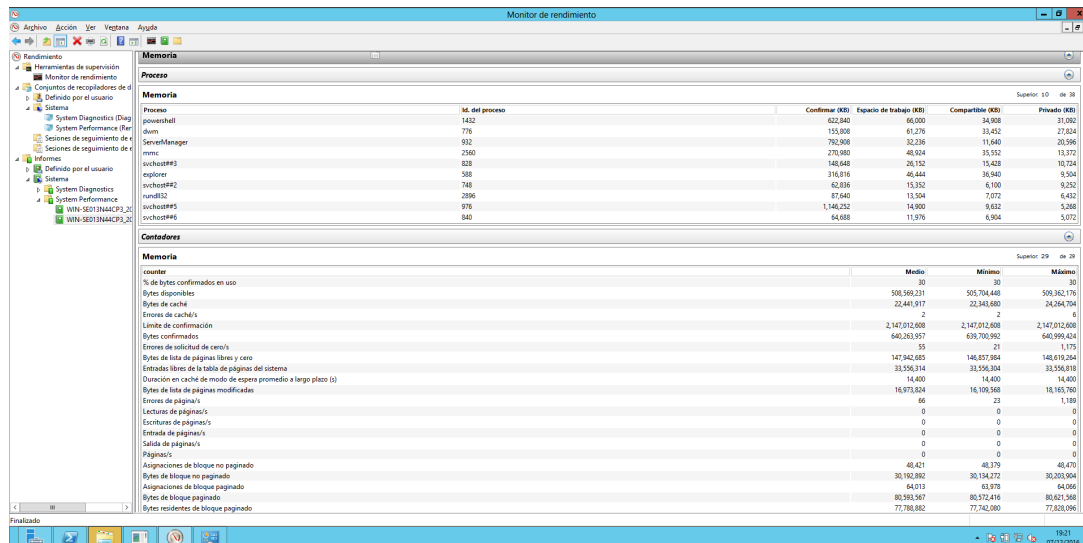


Figura 4.3: Resultado de memoria del monitor "System Performance" en Windows Server.

Si nos centramos precisamente en el uso de memoria, vemos como gran parte de esta se encuentra ocupada por el procesos "svchost", el cual gestiona diversos servicios de Windows, y por la aplicación "ServerManager" o Administrador del servidor.

5. Cuestión 5

5.1. Cree un recopilador de datos definido por el usuario (modo avanzado) con perfmon 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.

Para la creación, se siguen los pasos especificados por Microsoft en su página de manuales Technet [12].

Nos vamos al submenú "Definidos por el usuario" dentro de "Conjunto de recopiladores de datos", hacemos click derecho y pinchamos en "Nuevo" - "Conjuntos de recopiladores de datos".

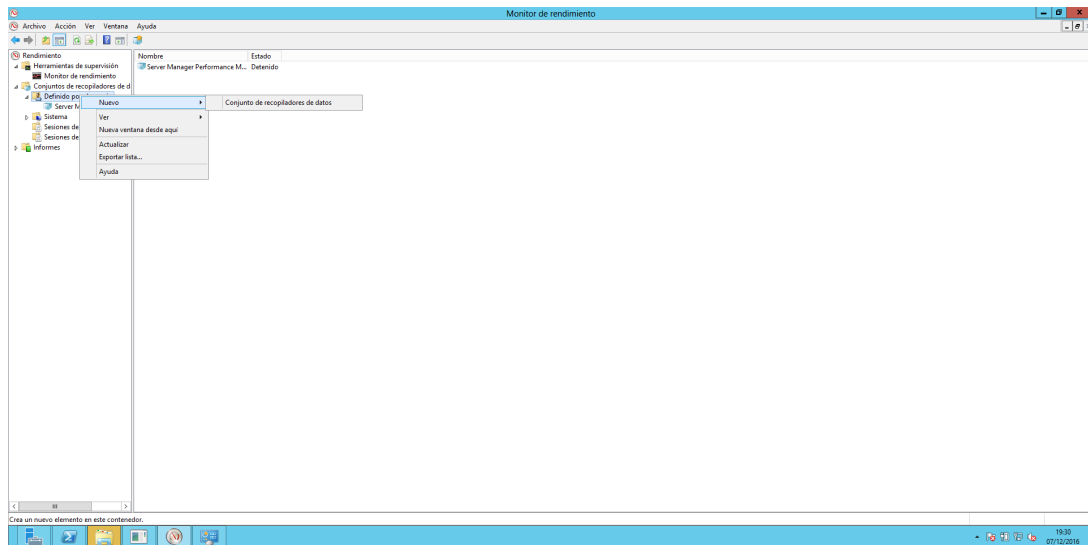


Figura 5.1: Creando nuevo conjunto de recopiladore de datos.

Se abrirá una interfaz gráfica para su creación. Aquí tendremos que darle un nombre y marcar la opción "Crear manualmente (avanzado)".

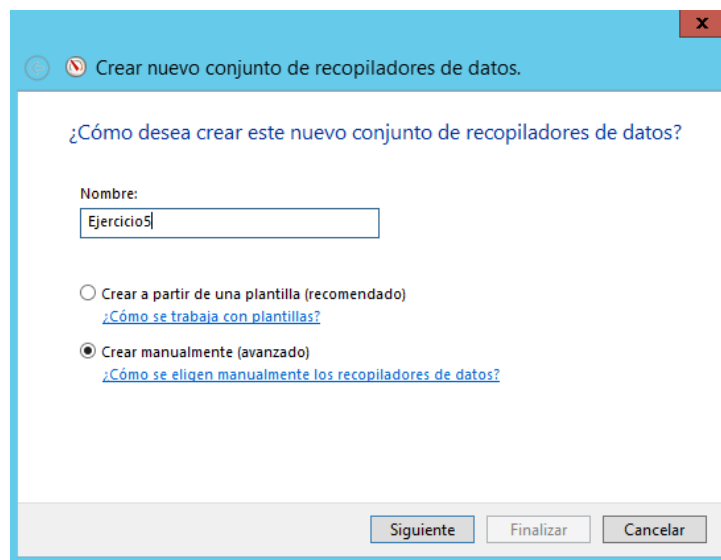


Figura 5.2: Interfaz para la creación de conjuntos de recopiladores de datos.

A continuación, seleccionamos los tipos de datos que deseamos incluir. En nuestro caso, "Contador de rendimiento" y "Datos de seguimiento de eventos".

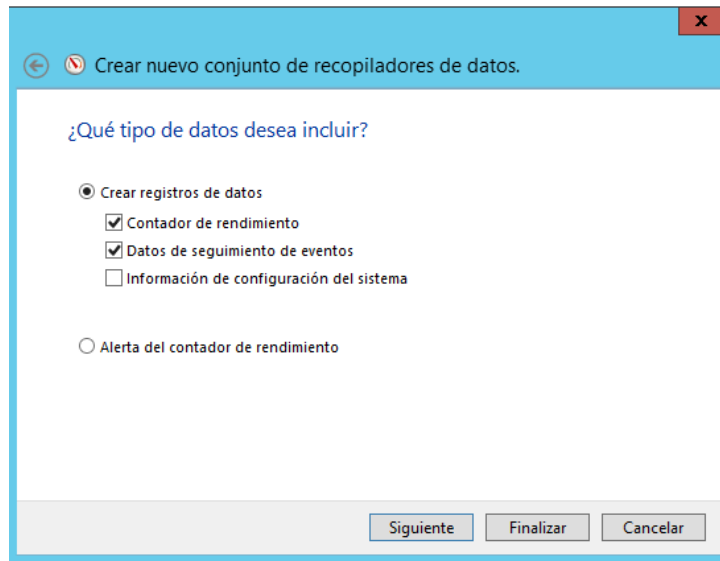


Figura 5.3: Selección de tipo de datos.

Ahora debemos de seleccionar los contadores de rendimiento que queremos registrar. Pinchamos en "Agregar" y se abrirá una ventana auxiliar donde tendremos que seleccionarlos. En este ejercicio se piden todos los correspondientes a procesador, proceso y servicio web.

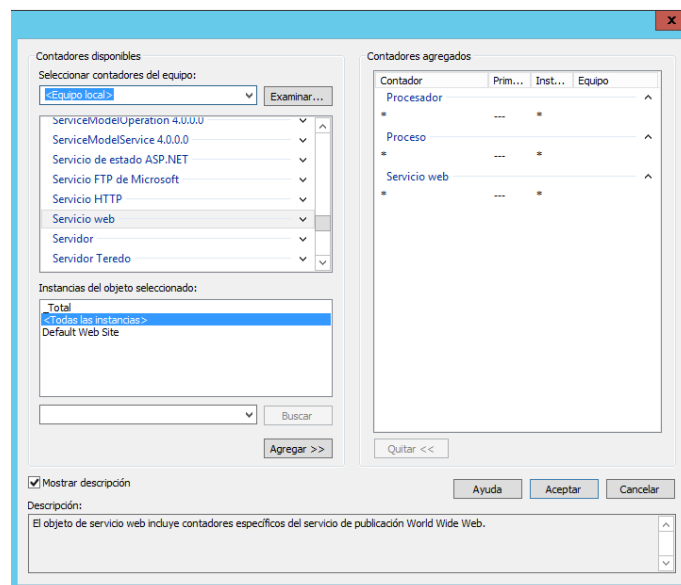


Figura 5.4: Selección de contadores de rendimiento a registrar.

Cuando terminemos de escoger los contadores de rendimiento que deseemos, estable-

ceamos el intervalo de muestra a 15 segundos tal y como se nos pide.

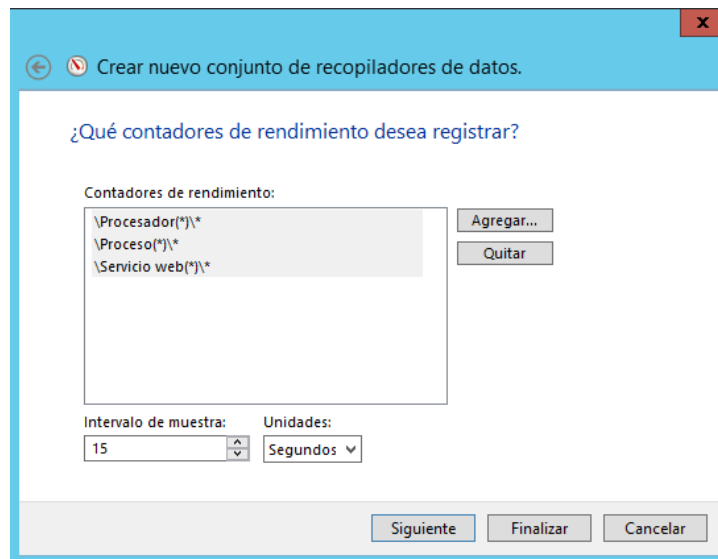


Figura 5.5: Confirmación de contadores de rendimiento a registrar.

Para guardar los datos en el path exigido, tendremos que añadirlo manualmente tal y como se muestra en la captura.

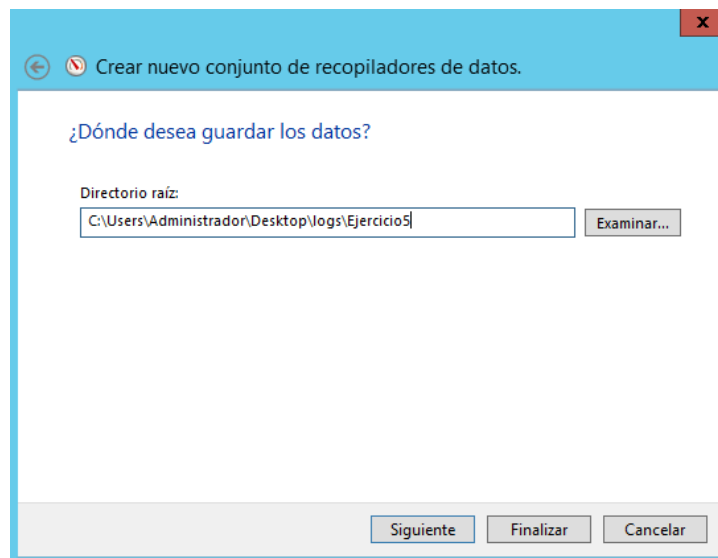


Figura 5.6: Selección de ubicación de guardado.

Se nos preguntará también qué usuario debe de ejecutarlo y la acción a realizar una vez termine el proceso de creación. En el ejercicio no se pide la ejecución de este, por lo

cual se marca "Guardar y cerrar".

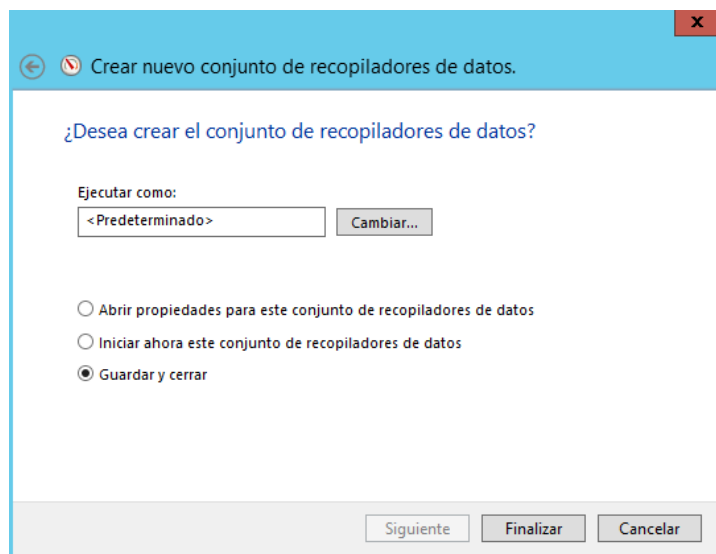


Figura 5.7: Paso final para la creación del conjunto de recopiladores de datos.

Una vez creado, podemos observarlo y ejecutarlo en el submenú "Definidos por el usuario".

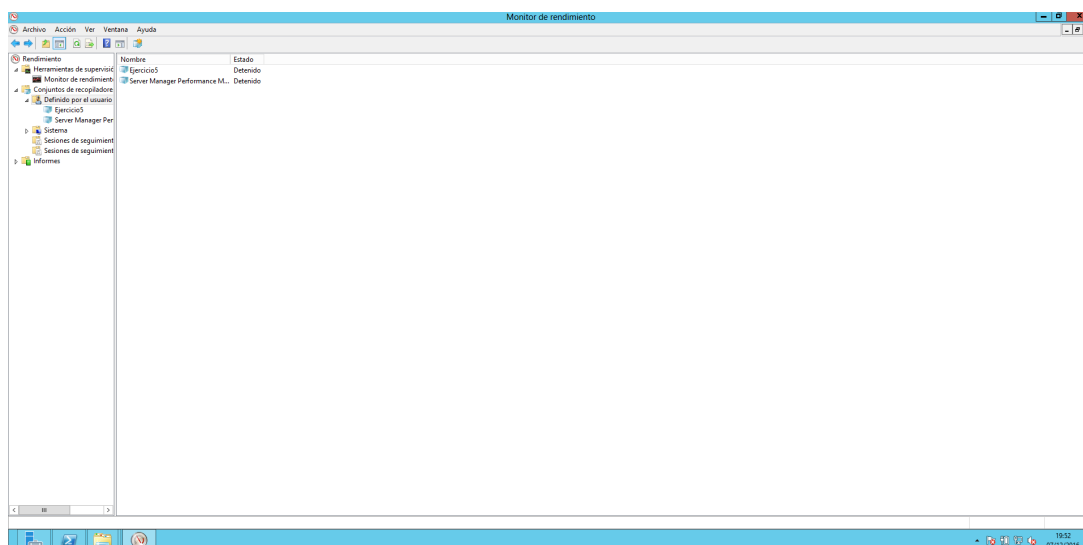


Figura 5.8: Resultado de memoria del monitor "System Performance" en Windows Server.

6. Cuestión 6

- 6.1. 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 demo especificada en el enunciado del ejercicio y comprobamos en primer lugar la latencia del disco por dispositivo del servidor monitorizado [13].

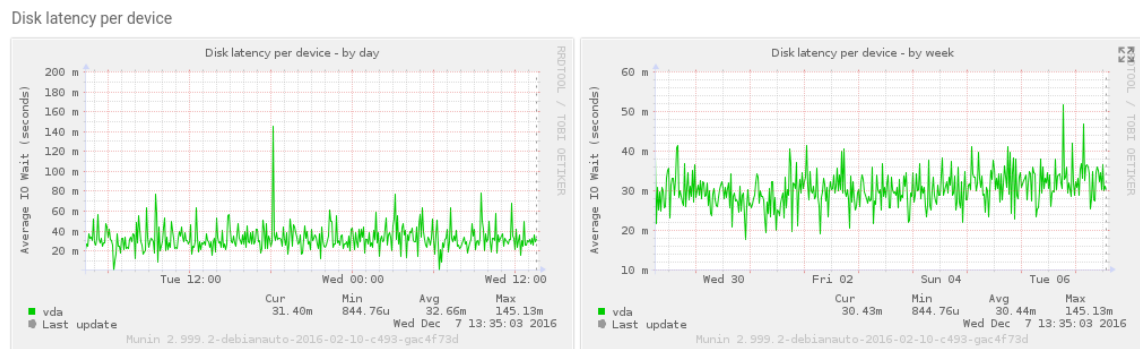


Figura 6.1: Latencia de disco por dispositivo por día y semana.

Comparando la gráfica diaria y semanal podemos observar como la latencia media del disco es mayor en las últimas 24 horas (a partir de la realización del ejercicio el día 07/12/16) comparada con la misma a lo largo de la semana. Esto quiere decir que la carga sobre el disco es mayor en el día medido o que existen factores externos que hacen que la latencia sea mayor. También se puede ver a simple vista un pico máximo en la gráfica diaria correspondiente al día 06/12/16 a las 18:00 horas aproximadamente, el cual podría indicar una subida repentina en el número de peticiones a disco. Por último, se puede divisar en la gráfica semanal como la media de latencia de disco por dispositivo ha subido a lo largo de la semana de manera progresiva, lo cual podría interpretarse como una subida gradual de la carga en este.

Se decide monitorizar otro parámetro, en concreto el número de hilos en ejecución [14].

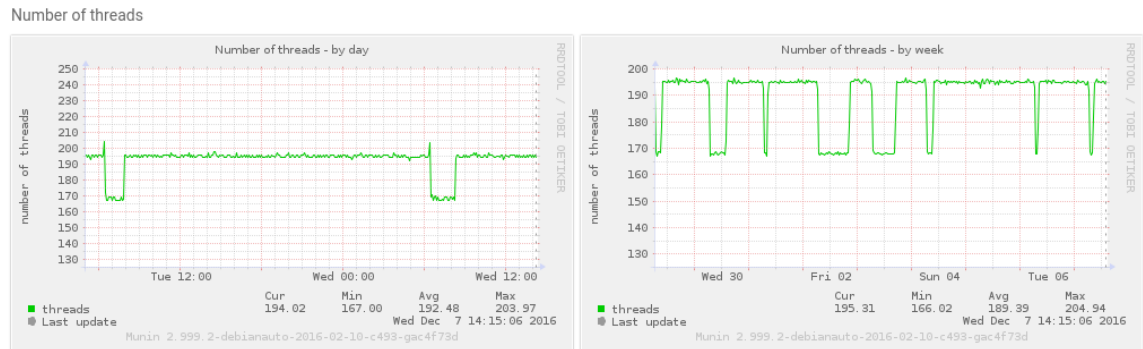


Figura 6.2: Número de hilos en ejecución por día y semana.

Comparamos ambas gráficas y vemos como en las últimas 24 horas (respecto a la realización del ejercicio el día 07/12/16) el número medio de hilos es mayor que la media semanal, lo cual podría explicar el aumento de latencia media de disco por dispositivo en el último día. También se ve a simple vista como cae lógicamente el número de hilos en la madrugada (entre 06:00 y 08:30 aproximadamente). Por último, en la gráfica semanal vemos como estas caídas nocturnas en el número de hilos son más acusadas cuando tienen lugar en el fin de semana, lo cual hace pensar que el servidor monitorizado tiene fines laborales.

7. Cuestión 7

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

Para responder esta cuestión se ha escogido el segundo de los artículos ofrecidos en el guión de prácticas [15]. En él, se explica la importancia del comando *strace* (system trace, "traza del sistema") para administradores de sistemas, pues este monitoriza todas las llamadas y señales del sistema realizadas por un proceso en concreto. El autor recalca la importancia de esta herramienta, ya que nos podemos ver envueltos en casos en que no encontremos un error que tuvo lugar en una aplicación y los archivos logs no muestren nada fuera de lo normal. Además, comenta que no hace falta tener conocimientos avanzados para el uso de esta herramienta más allá de los conocimientos requeridos para el uso de bash, pues la mayoría de llamadas al sistema que encontraremos serán como estas.

Para ejemplificar su uso, Latham [16], autor del artículo, nos muestra varios casos de uso. En el primero de ellos, monitoriza la ejecución del programa *cat* con un fichero que contiene únicamente dos líneas de texto. Para ello, muestra la salida que daría la

herramienta *strace* aplicado sobre el comando *cat file.txt* y explica varias de las llamadas al sistema, como *read* u *open*.

Muestra un segundo ejemplo en el cual el objetivo es encontrar la localización de un archivo log de un servicio, en concreto apache. Para ello, usa el comando *strace* con las opciones *-Ff*, la cual le permite monitorizar las llamadas al sistema producidas por procesos hijos; *-o <fichero>* para guardar la salida de *strace* en un fichero especificado; y *-e <llamada al sistema>* para observar únicamente las llamadas al sistema que coincidan con la dada (en este caso, *open*). Reinicia el servicio junto con *strace* y encuentra los ficheros logs de este.

El tercer ejemplo que encontramos en el artículo, trata de encontrar los fallos de una aplicación que no son registrados en los ficheros logs, ya que, como ocurre en este caso, el servicio no ha podido si quiera ejecutarse. Para ello, ejecuta *strace* con los mismos parámetros que en el ejemplo anterior y comprueba que no se puede abrir uno de los ficheros requeridos.

8. Cuestión 8

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

Para la realización de este ejercicio se ha decidido crear un script en Python usando uno de los profilers presentados en el guión de prácticas [17]

El script creado calcula los 100000 primeros números primos y los muestra por pantalla. Durante esta ejecución, un profiler recogerá datos sobre el comportamiento del script y los mostrará también por la salida estándar. El código del script es el siguiente:

```
import cProfile, pstats, StringIO
import math
import sys

#####
def obtener_primos(limite):
    primos=[2]
    for num in range(3,limite+1,2):
        if all(num%i!=0 for i in range(2,int(math.sqrt(num))+1)):
            primos.append(num)

    return primos
#####
pr = cProfile.Profile()
pr.enable()
```



```
primos=obtener_primos(1000000)
print(primos)
print "\n\n—————\n\n"

pr.disable()
s = StringIO.StringIO()
sortby = 'cumulative'
ps = pstats.Stats(pr, stream=s).sort_stats(sortby)
ps.print_stats()
print s.getvalue()
```

Como observamos en los resultados del profiler, se realizan 2905285 llamadas a funciones en 0.741 segundos. El orden de las entradas en la tabla dependerá del tiempo acumulado de cada tipo de llamada, siendo de mayor a menor. Podemos observar que se realiza una única llamada a la función principal del script, la cual calcula la lista de números primos. El tiempo acumulado de esta supone prácticamente el tiempo total del script. La segunda entrada de la tabla se trata de la función incorporada *all* y la tercera la sentencia 9 del script, la cual tiene el mayor número de llamadas y tiempo total. El resto de llamadas poseen un tiempo acumulado insignificante.

```

97501, 97511, 97523, 97547, 97549, 97553, 97561, 97571, 97577, 97579, 97583, 97607, 97609, 97613, 97
649, 97651, 97673, 97687, 97711, 97729, 97771, 97777, 97787, 97789, 97813, 97829, 97841, 97843, 9784
7, 97849, 97859, 97861, 97871, 97879, 97883, 97919, 97927, 97931, 97943, 97961, 97967, 97973, 97987,
98009, 98011, 98017, 98041, 98047, 98057, 98081, 98101, 98123, 98129, 98143, 98179, 98207, 98213, 9
8221, 98227, 98251, 98257, 98269, 98297, 98299, 98317, 98321, 98323, 98327, 98347, 98369, 98377, 983
87, 98389, 98407, 98411, 98419, 98429, 98443, 98453, 98459, 98467, 98473, 98479, 98491, 98507, 98519
, 98533, 98543, 98561, 98563, 98573, 98597, 98621, 98627, 98639, 98641, 98663, 98669, 98689, 98711,
98713, 98717, 98729, 98731, 98737, 98773, 98779, 98801, 98807, 98809, 98837, 98849, 98867, 98869, 98
873, 98887, 98893, 98897, 98899, 98909, 98911, 98927, 98929, 98939, 98947, 98953, 98963, 98981, 9899
3, 98999, 99013, 99017, 99023, 99041, 99053, 99079, 99083, 99089, 99103, 99109, 99119, 99131, 99133,
99137, 99139, 99149, 99173, 99181, 99191, 99223, 99233, 99241, 99251, 99257, 99259, 99277, 99289, 9
9317, 99347, 99349, 99367, 99371, 99377, 99391, 99397, 99401, 99409, 99431, 99439, 99469, 99487, 994
97, 99523, 99527, 99529, 99551, 99559, 99563, 99571, 99577, 99581, 99607, 99611, 99623, 99643, 99661
, 99667, 99679, 99689, 99707, 99709, 99713, 99719, 99721, 99733, 99761, 99767, 99787, 99793, 99809,
99817, 99823, 99829, 99833, 99839, 99859, 99871, 99877, 99881, 99901, 99907, 99923, 99929, 99961, 99
971, 99989, 999911

-----

2905285 function calls in 0.741 seconds

Ordered by: cumulative time

ncalls  tottime  percall  cumtime  percall filename:lineno(function)
1        0.073    0.073    0.741    0.741 profiler.py:6(obtener_primos)
49999    0.259    0.000    0.620    0.000 {all}
2745694  0.366    0.000    0.366    0.000 profiler.py:9(<genexpr>)
50000    0.035    0.000    0.035    0.000 {range}
49999    0.007    0.000    0.007    0.000 {math.sqrt}
9591     0.001    0.000    0.001    0.000 {method 'append' of 'list' objects}
1        0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}

[07/12/16 gmm@ubuntuserver:~] $ _

```

Figura 8.1: Resultado de la ejecución del script profiler en python.

9. Cuestión 9

9.1. 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).

La realización de esta cuestión se hará mediante phpMyAdmin, con el cual se ha creado previamente una tabla y se han insertado varias tuplas. Con la tabla en correcto estado, tendremos que activar el profiler para las consultas realizadas con esta herramienta tal y como explica Envato en uno de sus manuales [18], activando la casilla "Perfilando" dentro del menú "Examinar".

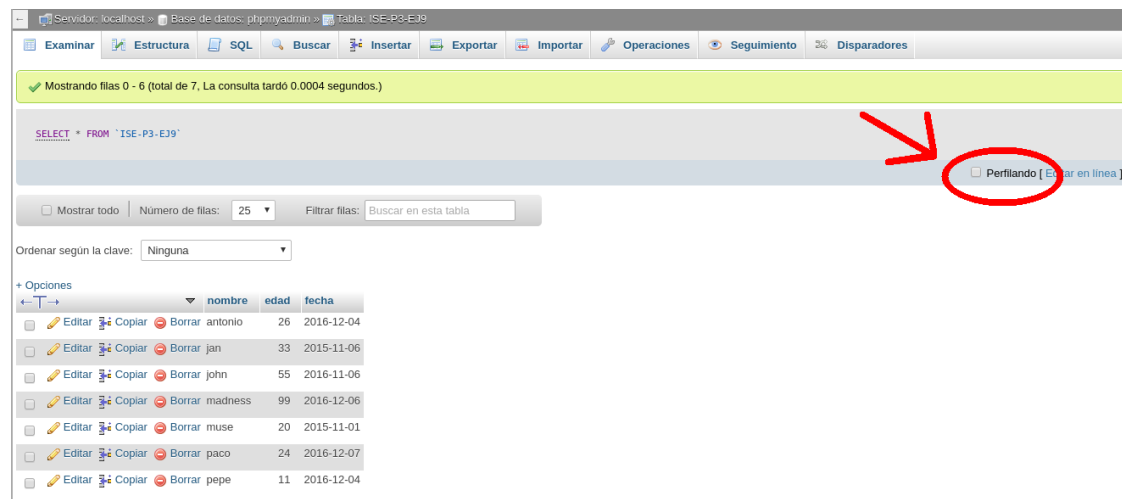


Figura 9.1: Activación del profiler en phpMyAdmin.

Una vez activada dicha casilla, podremos proceder a realizar cualquier consulta para probar la funcionalidad del profiler. En este caso, vamos a consultar todas las ocurrencias de la columna "nombre" de la tabla creada previamente con nombre "ISE-P3-EJ9".



Figura 9.2: Consulta sobre la tabla creada con phpMyAdmin.

Realizada la consulta, obtendremos el resultado de esta y los datos capturados por el profiler.

Perfilando							
Perfilación detallada			Resumen por estado				
Orden	Estado	Tiempo	Estado	Tiempo Total	% de Tiempo	Llamadas	Ø de Tiempo
1	Starting	24 µs	Starting	24 µs	22.86%	1	24 µs
2	Checking Permissions	4 µs	Checking Permissions	4 µs	3.81%	1	4 µs
3	Opening Tables	11 µs	Opening Tables	11 µs	10.48%	1	11 µs
4	Init	6 µs	Init	6 µs	5.71%	1	6 µs
5	System Lock	4 µs	System Lock	4 µs	3.81%	1	4 µs
6	Optimizing	2 µs	Optimizing	2 µs	1.90%	1	2 µs
7	Statistics	5 µs	Statistics	5 µs	4.76%	1	5 µs
8	Preparing	4 µs	Preparing	4 µs	3.81%	1	4 µs
9	Executing	1 µs	Executing	1 µs	0.95%	1	1 µs
10	Sending Data	27 µs	Sending Data	27 µs	25.71%	1	27 µs
11	End	2 µs	End	2 µs	1.90%	1	2 µs
12	Query End	3 µs	Query End	3 µs	2.86%	1	3 µs
13	Closing Tables	3 µs	Closing Tables	3 µs	2.86%	1	3 µs
14	Freeing Items	5 µs	Freeing Items	5 µs	4.76%	1	5 µs
15	Cleaning Up	4 µs	Cleaning Up	4 µs	3.81%	1	4 µs

✓ Mostrando filas 0 - 6 (total de 7, La consulta tardó 0.0002 segundos.)

Figura 9.3: Datos obtenidos por el profiler sobre la consulta realizada con phpMyAdmin.

✓ Mostrando filas 0 - 6 (total de 7, La consulta tardó 0.0002 segundos.)

```
SELECT nombre FROM `ISE-P3-EJ9`
```

☐ Mostrar todo | Número de filas: 25 | Filtrar filas:

Ordenar según la clave: Ninguna

+ Opciones

☐ Editar ☐ Copiar ☐ Borrar antonio

☐ Editar ☐ Copiar ☐ Borrar jan

☐ Editar ☐ Copiar ☐ Borrar john

☐ Editar ☐ Copiar ☐ Borrar madness

☐ Editar ☐ Copiar ☐ Borrar muse

☐ Editar ☐ Copiar ☐ Borrar paco

☐ Editar ☐ Copiar ☐ Borrar pepe

☐ Seleccionar todo Para los elementos que están marcados: ☐ Editar ☐ Copiar ☐ Borrar ☐ Exportar

☐ Mostrar todo | Número de filas: 25 | Filtrar filas:

Figura 9.4: Resultado de la consulta realizada con phpMyAdmin.

Como podemos comprobar en los resultados de profiler, la mayoría del tiempo tomado para la consulta se encuentra en la inicialización, el envío de datos y la apertura de la tabla. Estas llamadas serían las que habría que mejorar para obtener un mejor rendimiento en las consultas, ya sea por software o por hardware en el servidor.

Referencias

- [1] [dpkg - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [2] [apt - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [3] [How do I show apt-get package management history via command line?](#) Referencia no válida. Consultado el 4 de diciembre de 2016.
- [4] [zless - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [5] [cron - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [6] [crontab - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [7] [date - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [8] [dmesg - Linux man page](#). Consultado el 4 de diciembre de 2016.
- [9] [SCSI Trade Association](#). Consultado el 4 de diciembre de 2016.
- [10] Microsoft Technet. [Windows Performance Monitor](#). Consultado el 7 de diciembre de 2016.
- [11] Microsoft Technet. [System Requirements and Installation Information for Windows Server 2012 R2](#). Consultado el 7 de diciembre de 2016.
- [12] Microsoft Technet. [Create a Data Collector Set Manually](#). Consultado el 7 de diciembre de 2016.
- [13] [Munin demo - Disk latency](#). Consultado el 7 de diciembre de 2016.
- [14] [Munin demo - Num threads](#). Consultado el 7 de diciembre de 2016.
- [15] Lee Latham. [Sysadmin Tips and Tricks - Using strace to Monitor System Calls](#). Consultado el 9 de diciembre de 2016.
- [16] [Lee Latham - SoftLayer Blog](#). Consultado el 9 de diciembre de 2016.
- [17] Python. [The Python Profilers](#). Consultado el 7 de diciembre de 2016.
- [18] Envato. [Profiling MySQL Queries with phpMyAdmin](#). Consultado el 8 de diciembre de 2016.