

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

Memoria Práctica 3

José Álvaro Garrido López

9 de diciembre de 2016

Índice

Cuestión 1.	4
a) ¿Qué archivo le permite ver qué programas se han instalado con el gestor de paquetes?	4
b) ¿Qué significan las terminaciones 1.gz o 2.gz de los archivos en ese directorio?	6
Cuestión 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)	7
Cuestión 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	9
Cuestión 4. Ejecute el monitor de "System Performance" y muestre el resultado. Incluya capturas de pantalla comentando la información que aparece	11
Cuestión 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	14
Cuestión 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.	20
Cuestión 7. Escriba un breve resumen sobre alguno de los artículos donde se muestra el uso de strace o busque otro y coméntelo	24
Cuestión 8. Escriba un script en Python o PHP y analice su comportamiento usando el profiler presentado	25
Cuestión 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)	27

Índice de figuras

0.1. Fragmento del fichero <i>/var/log/yum.log</i>	4
0.2. Captura de la interfaz gráfica de <i>gnome-system-log</i>	5

0.3. Captura de selección por fecha en <i>gnome-system-log</i>	5
0.4. Captura del archivo <i>/var/log/apt/history.log</i>	6
0.5. Captura del archivo <i>/var/log/dmes.1.gz</i> descomprimido	6
0.6. Captura del fichero <i>crontab</i>	8
0.7. Funcionamiento del fichero <i>crontab</i> editado	8
0.8. Ejecución del comando <i>dmesg</i> antes de introducir el <i>USB</i>	9
0.9. Ejecución del comando <i>dmesg</i> después de introducir el <i>USB</i>	10
0.10. Mensajes de diagnóstico segundos después	10
0.11. Inicio de System Performance	11
0.12. Informe de rendimiento del sistema en System Performance	12
0.13. Apartado <i>CPU</i> del informe	12
0.14. Información sobre el procesador en el informe	12
0.15. Servicios del sistema	13
0.16. Información sobre el uso de memoria en el sistema	13
0.17. Estadísticas del informe	14
0.18. Creación de un nuevo conjunto de <i>recopiladores de datos</i>	15
0.19. <i>Wizard</i> de creación del recopilador	16
0.20. Adición de contadores referentes al procesador, proceso y servicio web	17
0.21. Definición del intervalo de muestra para el recopilador	18
0.22. Almacenamiento de los datos en la carpeta <i>logs</i> del escritorio	18
0.23. Resultado del recopilador de datos	19
0.24. Identificadores del gráfico	19
0.25. Porcentaje de tiempo en modo privilegiado	20
0.26. Vista general de Munin	21
0.27. Gráficos de Munin , resultados por año	21
0.28. Gráfico sobre el número de hebras utilizadas	22
0.29. Gráfico sobre interrupciones del sistema	23
0.30. Gráfico sobre memoria utilizada	24
0.31. Resultado del <i>profile</i> de la primera versión	26
0.32. Resultado del <i>profile</i> de la segunda versión	27
0.33. Creación y configuración del <i>profiler</i>	28
0.34. Ejecución de SHOW PROFILES	29
0.35. Ejecución de SHOW PROFILE	29
0.36. <i>Profile</i> de la creación de una base de datos	29
0.37. <i>Profile</i> de una consulta	30

Cuestión 1.

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

En el caso de **CentOS**, como se menciona en [2], existe un fichero llamado *yum.log* donde se almacenan todas las acciones que se realizan sobre el comando *yum*. En 0.1 se puede ver un fragmento del log.

```
alvarogl nov 2016 > sudo cat /var/log/yum.log
[sudo] password for alvarogl:
Nov 02 17:15:29 Updated: openssh-6.6.1p1-25.el7_2.x86_64
Nov 02 17:15:31 Updated: openssh-server-6.6.1p1-25.el7_2.x86_64
Nov 02 17:15:31 Updated: openssh-clients-6.6.1p1-25.el7_2.x86_64
Nov 09 12:07:43 Installed: 1:perl-Error-0.17020-2.el7.noarch
Nov 09 12:07:43 Installed: perl-Thread-Queue-3.02-2.el7.noarch
Nov 09 12:07:50 Updated: glibc-2.17-106.el7_2.8.x86_64
Nov 09 12:08:19 Updated: glibc-common-2.17-106.el7_2.8.x86_64
Nov 09 12:08:25 Installed: libmpc-1.0.1-3.el7.x86_64
Nov 09 12:08:25 Installed: apr-1.4.8-3.el7.x86_64
Nov 09 12:08:26 Installed: m4-1.4.16-10.el7.x86_64
Nov 09 12:08:27 Installed: libquadmath-4.8.5-4.el7.x86_64
Nov 09 12:08:27 Installed: apr-util-1.5.2-6.el7.x86_64
Nov 09 12:08:29 Installed: patch-2.7.1-8.el7.x86_64
Nov 09 12:08:30 Installed: subversion-libs-1.7.14-10.el7.x86_64
Nov 09 12:08:31 Installed: libgfortran-4.8.5-4.el7.x86_64
Nov 09 12:08:38 Installed: cpp-4.8.5-4.el7.x86_64
Nov 09 12:08:39 Installed: mokutil-0.9-2.el7.x86_64
Nov 09 12:08:39 Installed: perl-Data-Dumper-2.145-3.el7.x86_64
```

Figura 0.1: Fragmento del fichero */var/log/yum.log*

También, como se explica en el guión de prácticas, tenemos una alternativa con interfaz gráfica, *gnome-system-log*. Ejecutamos

gnome-system-log

desde terminal, y nos aparece la ventana que se observa en 0.2.

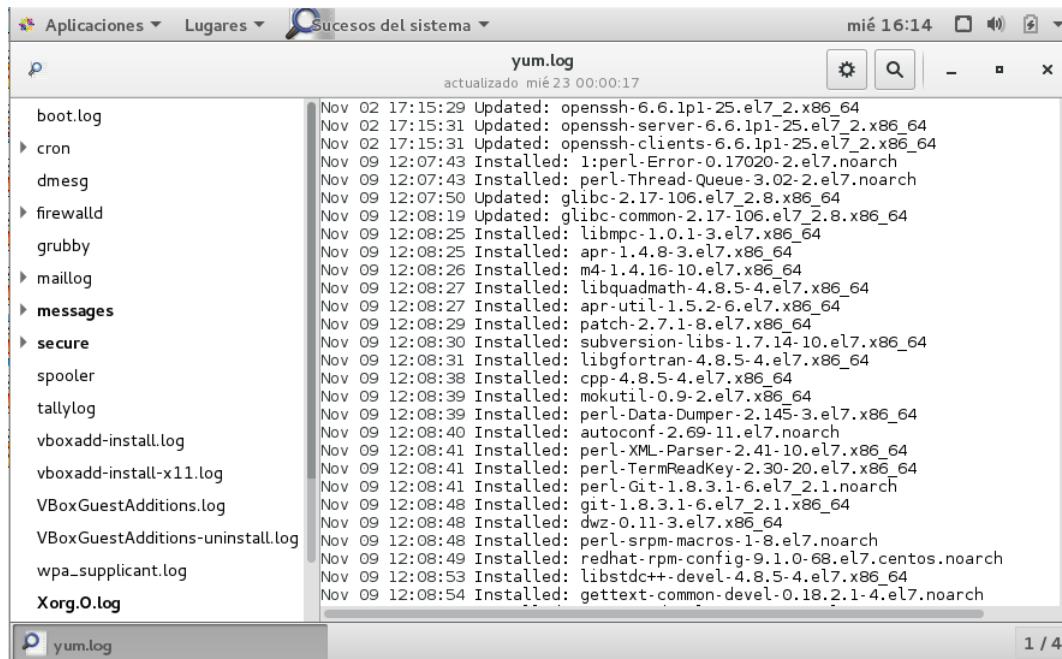


Figura 0.2: Captura de la interfaz gráfica de *gnome-system-log*

Si desplegamos en la interfaz el fichero *yum.log*, como se muestra en 0.3, podemos comprobar que el programa nos ordena por fecha las entradas del fichero, lo cual puede ser de gran utilidad.

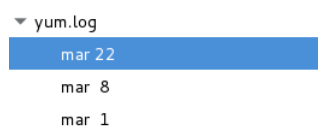


Figura 0.3: Captura de selección por fecha en *gnome-system-log*

En el caso de **Ubuntu**, en [1], se nos explica que existen varios ficheros históricos que almacenan la actividad de *apt*, como lo son */var/log/dpkg.log*, */var/log/apt/term.log* y */var/log/aptitude*. El más conciso de ellos es */var/log/apt/term.log*, aunque también existe otro archivo que se menciona en [15], */var/log/apt/history.log*, cuyo contenido podemos ver en 0.4 que almacena los comandos ejecutados sobre *apt*.

```

Start-Date: 2016-11-25 16:14:40
Commandline: apt install openssl-server
Install: ncurses-term:amd64 (5.9-20140118-1ubuntu1, automatic), python-urllib3:amd64 (1.7.1-1ubuntu4, automatic), openssl-server:amd64 (6.6p1-2ubuntu2.8), openssl-sftp-server:amd64 (6.6p1-2ubuntu2.8, automatic), ssh-import-id:amd64 (3.21-0ubuntu1, automatic), python-requests:amd64 (2.2.1-1ubuntu0.3, automatic), libck-connector0:amd64 (0.4.5-3.1ubuntu2, automatic), libwrap0:amd64 (7.6.q-25, automatic), tcpd:amd64 (7.6.q-25, automatic)
Upgrade: openssl-client:amd64 (6.6p1-2ubuntu2.7, 6.6p1-2ubuntu2.8)
End-Date: 2016-11-25 16:15:04

Start-Date: 2016-11-25 16:17:19
Commandline: apt install nmap
Install: libblas3:amd64 (1.2.20110419-7, automatic), liblinear1:amd64 (1.8+dfsg-1ubuntu1, automatic), nmap:amd64 (6.40-0.2ubuntu1), liblua5.2-0:amd64 (5.2.3-1, automatic), liblinear-tools:amd64 (1.8+dfsg-1ubuntu1, automatic)
End-Date: 2016-11-25 16:17:24

```

Figura 0.4: Captura del archivo `/var/log/apt/history.log`

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

Podemos descomprimir estos archivos ejecutando el siguiente comando:

```
gunzip /var/log/archivo.gz
```

En [5] se explica el uso del comando `gunzip`.

En [16] se habla del problema que acarrea el tamaño de los logs del sistema, y que una opción es comprimir estos archivos en formato `.gz`.

Podemos ver un fragmento del archivo `/var/log/dmesg.1.gz` descomprimido en 0.5.

```

GNU nano 2.2.6      File: /var/log/dmesg.1      Modified
[  80.368908] intel_rapl: no valid rapl domains found in package 0
[  80.923757] snd_intel8x0 0000:00:05.0: white list rate for 1020:0177 is 40000
[  80.925585] [drm] VRAM 01000000
[  80.931520] [TTM] Zone kernel: Available graphics memory: 508232 KiB
[  80.931523] [TTM] Initializing pool allocator
[  80.931530] [TTM] Initializing DMA pool allocator
[  80.932202] checking generic (e0000000 130000) vs hw (e0000000 1000000)
[  80.932205] fb: switching to vboxdrmfb from UESA UGA
[  80.932257] Console: switching to colour dummy device 80x25
[  80.932646] fbcon: vboxdrmfb (fb0) is primary device
[  80.934094] Console: switching to colour frame buffer device 100x37
[  80.936343] vboxvideo 0000:00:02.0: fb0: vboxdrmfb frame buffer device
[  80.936346] [drm] Initialized vboxvideo 1.0.0 20130823 for 0000:00:02.0 on minor 0
[  80.943534] ppdev: user-space parallel port driver
[  81.156944] audit: type=1400 audit(1480086154.837:2): apparmor="STATUS" operation="profile_load"
[  81.156952] audit: type=1400 audit(1480086154.837:3): apparmor="STATUS" operation="profile_load"
[  81.156955] audit: type=1400 audit(1480086154.837:4): apparmor="STATUS" operation="profile_load"
[  81.157140] audit: type=1400 audit(1480086154.837:5): apparmor="STATUS" operation="profile_replac
[  81.157144] audit: type=1400 audit(1480086154.837:6): apparmor="STATUS" operation="profile_replac
[  81.157289] audit: type=1400 audit(1480086154.837:7): apparmor="STATUS" operation="profile_replac
[  81.157431] audit: type=1400 audit(1480086154.837:8): apparmor="STATUS" operation="profile_replac
[  81.157436] audit: type=1400 audit(1480086154.837:9): apparmor="STATUS" operation="profile_replac
[  81.157440] audit: type=1400 audit(1480086154.837:10): apparmor="STATUS" operation="profile_replac
[  81.157861] audit: type=1400 audit(1480086154.837:11): apparmor="STATUS" operation="profile_replac
[  81.962479] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[  81.963329] e1000: eth0 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[  81.963628] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link becomes ready
[  83.211082] floppy0: no floppy controllers found
[  88.877660] EXT4-fs (dm-6): mounted filesystem with ordered data mode. Opts: (null)
[  89.126065] init: failsafe main process (1169) killed by TERM signal
[  89.515577] audit_printk_skb: 6 callbacks suppressed
[  89.515580] audit: type=1400 audit(1480086163.814:14): apparmor="STATUS" operation="profile_replac

```

Figura 0.5: Captura del archivo `/var/log/dmesg.1.gz` descomprimido

Cuestión 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`)

Como se explica en [11] y [3], través del comando:

```
sudo crontab -e <usuario>
```

podemos editar el fichero para programar tareas. Por defecto se nos asigna un archivo *crontab* en */tmp*, pero también tenemos acceso a */etc/cron.hourly*, */etc/cron.daily*, */etc/cron.weekly* y */etc/cron.monthly*.

Debemos añadir una línea con la siguiente sintaxis:

```
<minuto> <hora> <día del mes> <mes> <día de la semana> <comando>
```

En concreto, para lo que se nos pide, tendríamos que añadir las siguientes línea:

```
DATE=date + %F
59 23 * * * cp ~/codigo ~/seguridad/$($DATE)
```

En concreto, utilizamos la opción `+%F` de *date* para que el nombre de la carpeta sea del tipo *aaaa-mm-dd* y no contenga espacios, consultada la referencia [4] para comprender la sintaxis del comando.

Para probar el funcionamiento de lo que se pregunta en la cuestión, he añadido la tarea *cron* para que se ejecute cada minuto como se muestra en 0.6. 1

```

GNU nano 2.2.6      File: /tmp/crontab.HGDbOY/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# 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
DATE=date +%F
*/1 * * * * cp ~/codigo/* ~/seguridad/$(DATE)_

```

Figura 0.6: Captura del fichero *crontab*

En 0.7, se comprueba el funcionamiento de lo descrito en la cuestión.

```

alvarogl@ubuntuuserver:~dec 2016$ ls codigo/
codigo1 codigo2 codigo3 codigo4 codigo5 codigo6
alvarogl@ubuntuuserver:~dec 2016$ ls seguridad/2016-12-09/
alvarogl@ubuntuuserver:~dec 2016$ date
vie dic  9 15:27:29 CET 2016
alvarogl@ubuntuuserver:~dec 2016$ date
vie dic  9 15:28:14 CET 2016
alvarogl@ubuntuuserver:~dec 2016$ ls seguridad/2016-12-09/
codigo1 codigo2 codigo3 codigo4 codigo5 codigo6

```

Figura 0.7: Funcionamiento del fichero *crontab* editado

Cuestión 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

Según [14], para que se muestre la fecha y la hora de los mensajes de diagnóstico, debemos utilizar la opción `-F` del comando.

En 0.8 se muestran los mensajes de diagnóstico antes de introducir el USB y en 0.9 después de hacerlo.

```
alvarog1@ubuntu-server:~$ dmesg -T | tail
[mié dic 7 16:05:51 2016] init: failsafe main process (1211) killed by TERM signal
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.667:8): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/sbin/dhclient" pid=1542 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.667:9): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-client.action" pid=1542 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.667:10): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/lib/connman/scripts/dhclient-script" pid=1542 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.667:11): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/lib/NetworkManager/nm-dhcp-client.action" pid=1542 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.667:12): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/lib/connman/scripts/dhclient-script" pid=1542 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.667:13): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/lib/connman/scripts/dhclient-script" pid=1542 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.687:14): apparmor="STATUS" operation="profile_load" profile="unconfined" name="/usr/sbin/mysqld" pid=1544 comm="apparmor_parser"
[mié dic 7 16:05:52 2016] audit: type=1400 audit(1481123153.695:15): apparmor="STATUS" operation="profile_load" profile="unconfined" name="/usr/sbin/tcpdump" pid=1546 comm="apparmor_parser"
[mié dic 7 16:05:53 2016] audit: type=1400 audit(1481123154.227:16): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/sbin/mysqld" pid=1648 comm="apparmor_parser"
```

Figura 0.8: Ejecución del comando `dmesg` antes de introducir el *USB*

```

alvarogl@ubuntu-server:~$ dmesg -T | tail
[mié dic 7 16:08:20 2016] usb 1-2: new full-speed USB device number 3 using ohci-pci
[mié dic 7 16:08:20 2016] usb 1-2: New USB device found, idVendor=0bda, idProduct=0119
[mié dic 7 16:08:20 2016] usb 1-2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[mié dic 7 16:08:20 2016] usb 1-2: Product: USB2.0-CRW
[mié dic 7 16:08:20 2016] usb 1-2: Manufacturer: Generic
[mié dic 7 16:08:20 2016] usb 1-2: SerialNumber: 20090815198100000
[mié dic 7 16:08:20 2016] usb-storage 1-2:1.0: USB Mass Storage device detected
[mié dic 7 16:08:20 2016] scsi host4: usb-storage 1-2:1.0
[mié dic 7 16:08:20 2016] usbcore: registered new interface driver usb-storage
[mié dic 7 16:08:20 2016] usbcore: registered new interface driver uas

```

Figura 0.9: Ejecución del comando *dmesg* después de introducir el *USB*

Como podemos apreciar en 0.9, los mensajes de diagnóstico son sensibles a la conexión de un nuevo dispositivo por *USB*. Se indican entre otros parámetros, el fabricante, el número de serie y el nombre del dispositivo.

Si esperamos unos segundos nos aparece más información sobre el montaje del dispositivo. En 0.10 se especifican los bloques lógicos y el espacio en GB del dispositivo, la protección de escritura desactivada y otras cuestiones.

```

alvarogl@ubuntu-server:~$ dmesg -T | tail
[vie dic 9 16:22:43 2016] usbcore: registered new interface driver uas
[vie dic 9 16:22:44 2016] scsi 4:0:0:0: Direct-Access    Generic- SD/MMC          1.00 PQ: 0 ANSI
: 0 CCS
[vie dic 9 16:22:44 2016] sd 4:0:0:0: Attached scsi generic sg3 type 0
[vie dic 9 16:22:44 2016] sd 4:0:0:0: [sdcl] 7741440 512-byte logical blocks: (3.96 GB/3.69 GiB)
[vie dic 9 16:22:44 2016] sd 4:0:0:0: [sdcl] Write Protect is off
[vie dic 9 16:22:44 2016] sd 4:0:0:0: [sdcl] Mode Sense: 03 00 00 00
[vie dic 9 16:22:44 2016] sd 4:0:0:0: [sdcl] No Caching mode page found
[vie dic 9 16:22:44 2016] sd 4:0:0:0: [sdcl] Assuming drive cache: write through
[vie dic 9 16:22:44 2016]  sdc: sdc1
[vie dic 9 16:22:44 2016] sd 4:0:0:0: [sdcl] Attached SCSI removable disk

```

Figura 0.10: Mensajes de diagnóstico segundos después

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

Ejecutamos el **System Performance** como se muestra en 0.11

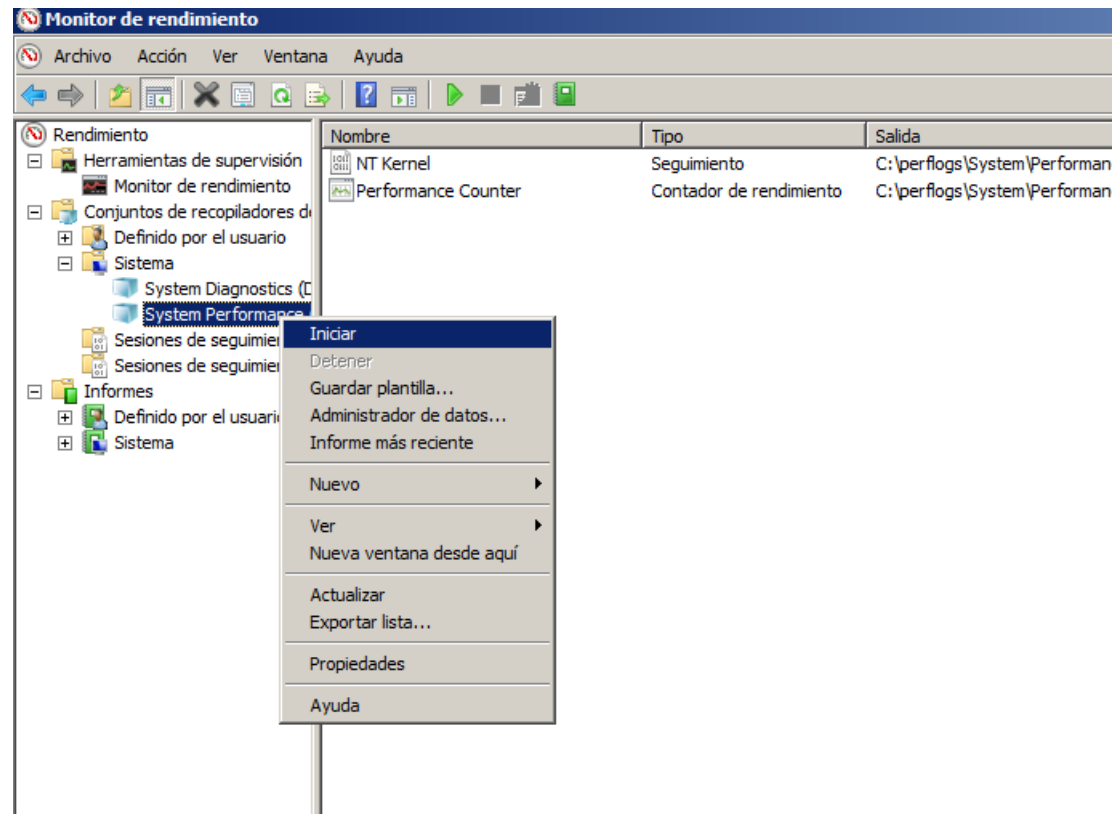


Figura 0.11: Inicio de **System Performance**

A continuación vemos el informe y algunos de los aspectos más llamativos. En 0.12 aparece la primera vista del informe.

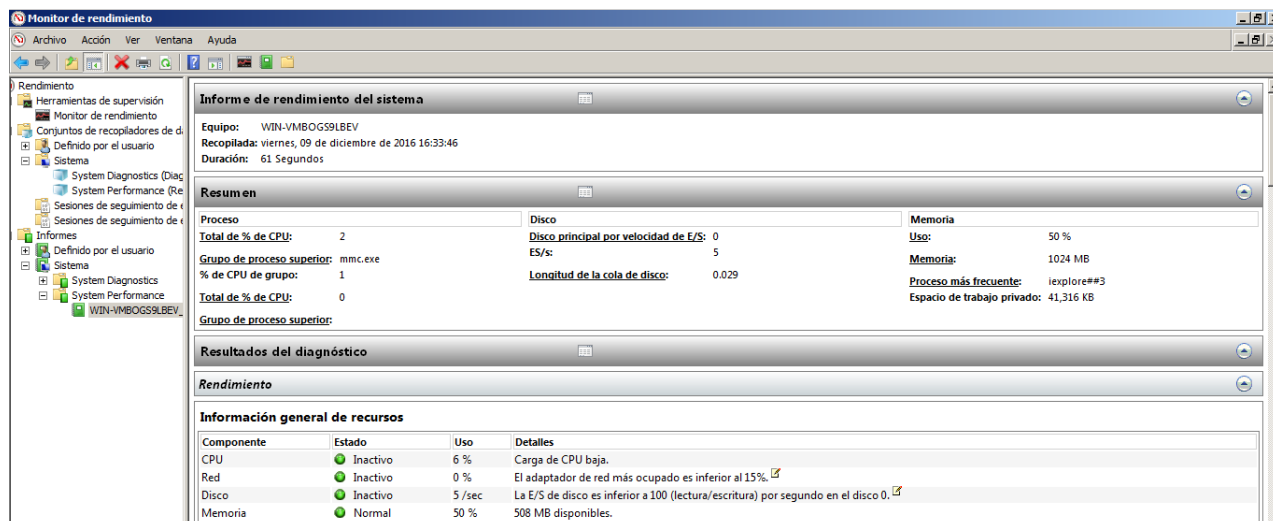


Figura 0.12: Informe de rendimiento del sistema en **System Performance**

En 0.13 se pueden ver cuáles son los procesos y en qué porcentaje usan a la CPU.

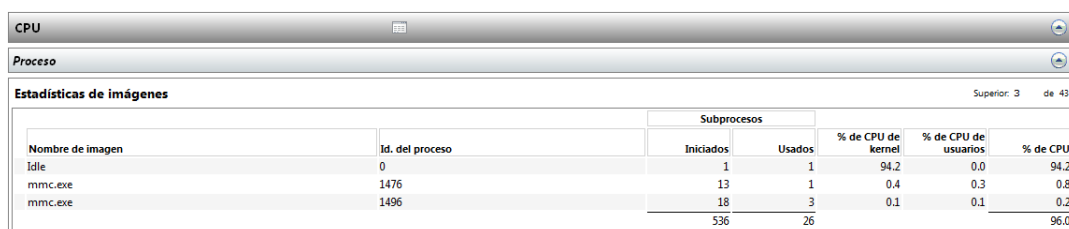


Figura 0.13: Apartado *CPU* del informe

En 0.14 se nos proporciona información referente al procesador. Aparecen el número medio de interrupciones, porcentaje de inactividad, porcentaje de ejecución de procesos en modo usuario y en modo privilegiado, etc.



Figura 0.14: Información sobre el procesador en el informe

En 0.15 se nos listan los servicios del sistema, el *processId*, y el porcentaje de CPU.

Servicios			Superior: 20 de 20
Proceso	processid	% de CPU	
spssvc.exe	1588	0.2	
svchost.exe	556	0.1	
svchost.exe	768	0.1	
lsass.exe	456	0.0	
svchost.exe	804	0.0	
svchost.exe	904	0.0	
msdtc.exe	1948	0.0	
spoolsv.exe	1036	0.0	
Servicios detenidos	-	0.0	
svchost.exe	1064	0.0	
svchost.exe	320	0.0	
svchost.exe	1088	0.0	
svchost.exe	852	0.0	
svchost.exe	1124	0.0	
svchost.exe	1920	0.0	
svchost.exe	1168	0.0	
svchost.exe	680	0.0	
svchost.exe	944	0.0	
TrustedInstaller.exe	592	0.0	
VBOSService.exe	616	0.0	

Figura 0.15: Servicios del sistema

En 0.16 aparecen información de los procesos en cuanto a la memoria que están utilizando.

Servicios			Superior: 20 de 20
Proceso	processid	% de CPU	
spssvc.exe	1588	0.2	
svchost.exe	556	0.1	
svchost.exe	768	0.1	
lsass.exe	456	0.0	
svchost.exe	804	0.0	
svchost.exe	904	0.0	
msdtc.exe	1948	0.0	
spoolsv.exe	1036	0.0	
Servicios detenidos	-	0.0	
svchost.exe	1064	0.0	
svchost.exe	320	0.0	
svchost.exe	1088	0.0	
svchost.exe	852	0.0	
svchost.exe	1124	0.0	
svchost.exe	1920	0.0	
svchost.exe	1168	0.0	
svchost.exe	680	0.0	
svchost.exe	944	0.0	
TrustedInstaller.exe	592	0.0	
VBOSService.exe	616	0.0	

Figura 0.16: Información sobre el uso de memoria en el sistema

En 0.17 se detallan estadísticas del informe, así como información del equipo, velocidad del procesador, cantidad de memoria, arquitectura del procesador y otros parámetros.

Estadísticas de informe	
Información del equipo	
Equipo:	WIN-VMBOGS9LBEV
Compilación de Windows:	7601
Procesadores:	1
Velocidad del procesador:	2594 MHz
Memoria:	1024 MB
Plataforma:	64 Bit
Información recopilada	
Hora de inicio:	viernes, 09 de diciembre de 2016 16:33:46
Hora de finalización:	viernes, 09 de diciembre de 2016 16:34:47
Duración:	61 Segundos
Búferes:	5
Eventos procesados:	2014
Eventos perdidos:	0
Eventos omitidos:	3
Usar ventana de sincronización:	Si

Figura 0.17: Estadísticas del informe

Cuestión 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

Siguiendo las indicaciones del enunciado de la cuestión, creamos un nuevo *recopilador de datos* como se muestra en 0.18.

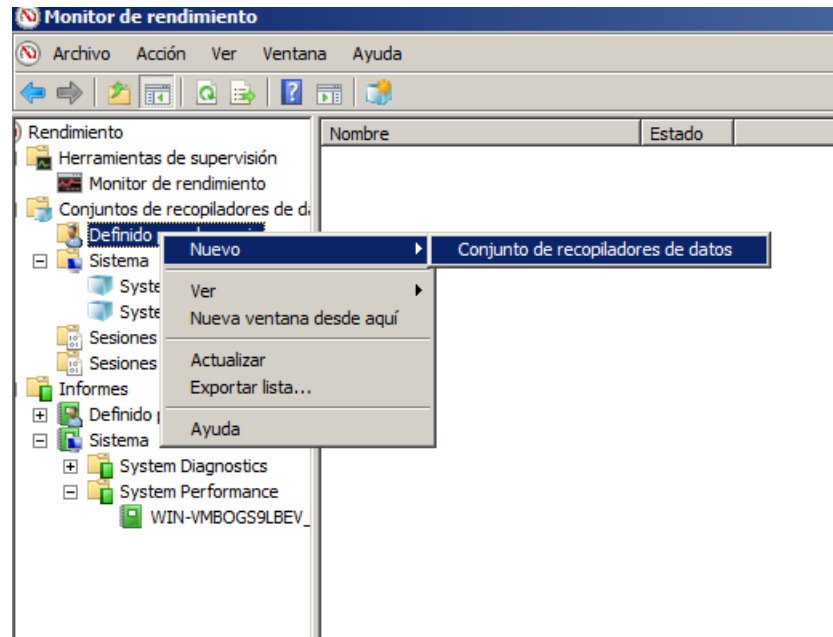


Figura 0.18: Creación de un nuevo conjunto de *recopiladores de datos*

Posteriormente, como se muestra en 0.19, seleccionamos la opción **Crear registro de datos**, marcamos el contador de rendimiento y los datos de seguimiento. La información de configuración del sistema es opcional.

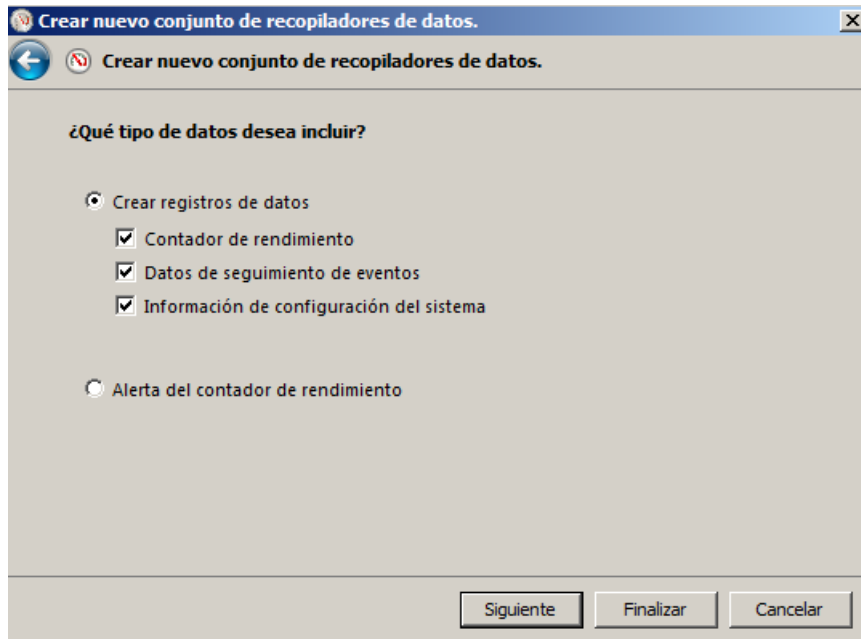


Figura 0.19: *Wizard* de creación del recopilador

Para agregar contadores, en el *Wizard*, como se ve en 0.20, buscamos el contador que deseemos y agregamos todas las instancias del objeto seleccionado. Al final deben quedar agregados los contadores referentes a procesador, proceso y servicio web.

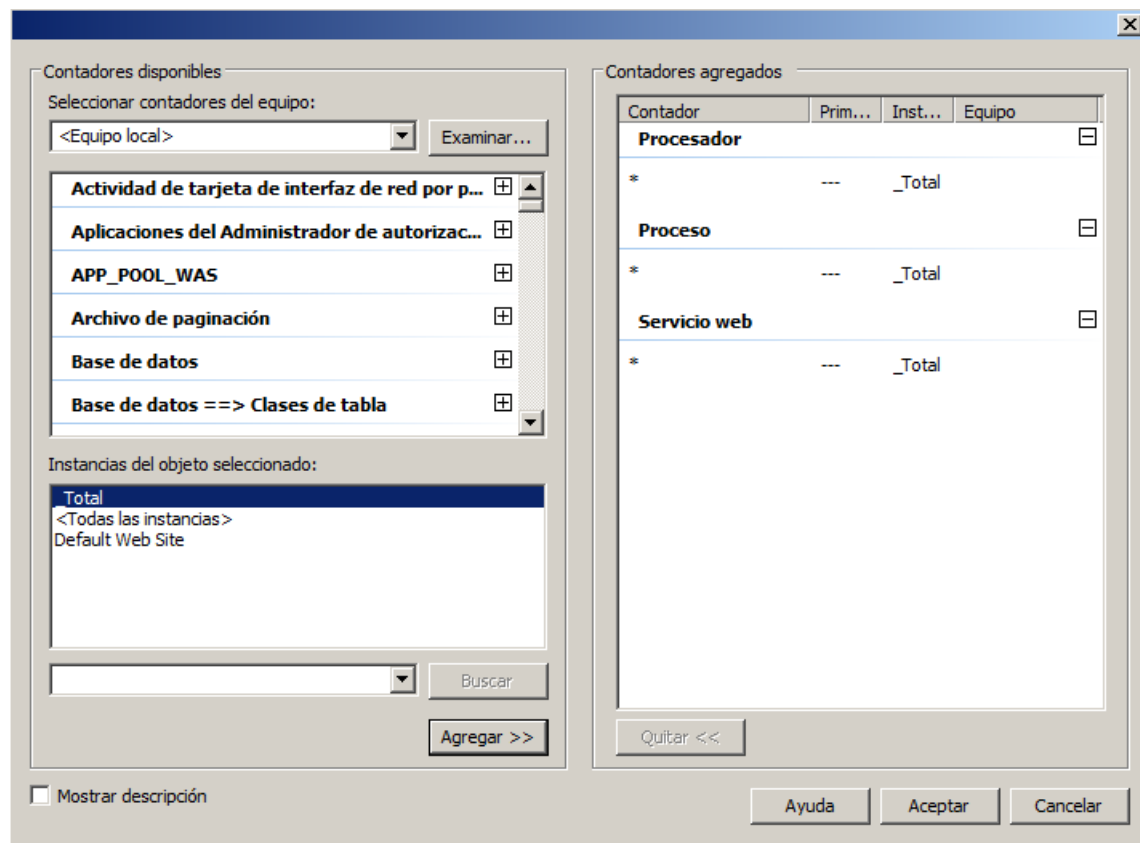


Figura 0.20: Adición de contadores referentes al procesador, proceso y servicio web

Para determinar el intervalo de muestra, en el paso referente a 0.21, indicamos el intervalo de 15 segundos.

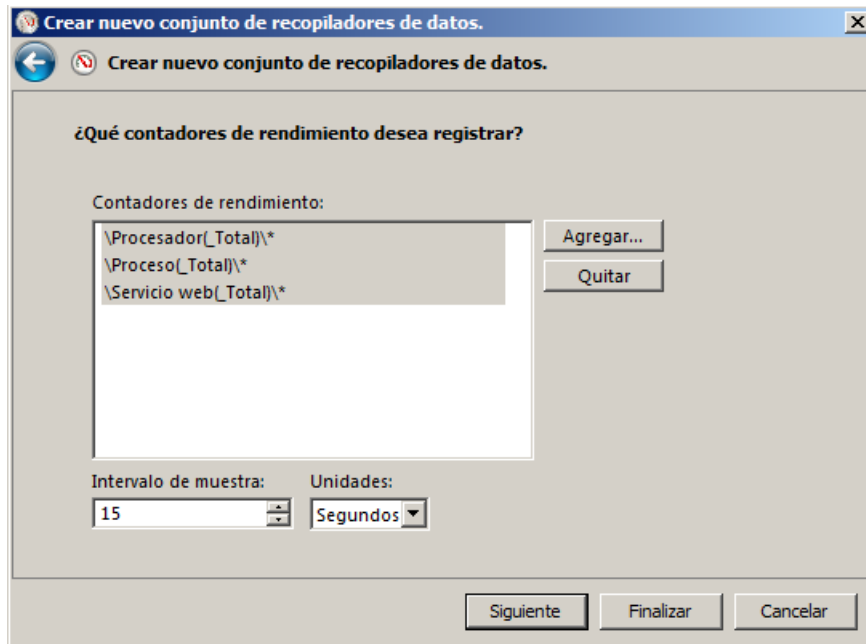


Figura 0.21: Definición del intervalo de muestra para el recopilador

Para terminar, almacenamos los datos en el */Desktop/logs*. Ver 0.22.

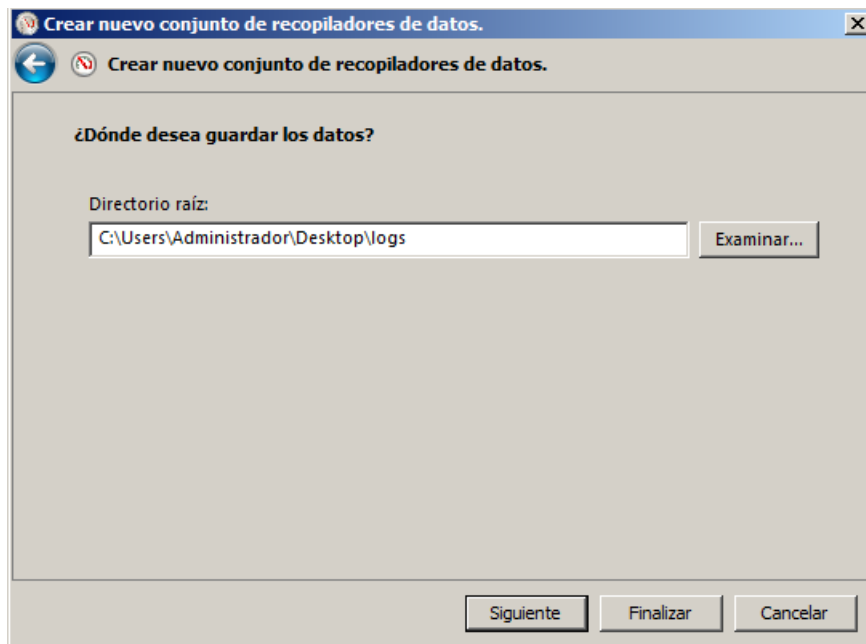


Figura 0.22: Almacenamiento de los datos en la carpeta *logs* del escritorio

Como se observa en los resultados 0.23, el informe ha recaudado un montón de datos de compleja interpretación. Si pinchamos en una zona del gráfico obtenido, nos aparecen debajo los colores identificadores de los datos que se encuentren en esa franja del gráfico. Por ejemplo, en 0.23, hay un pequeño pico en las operaciones de ES de dato/s.

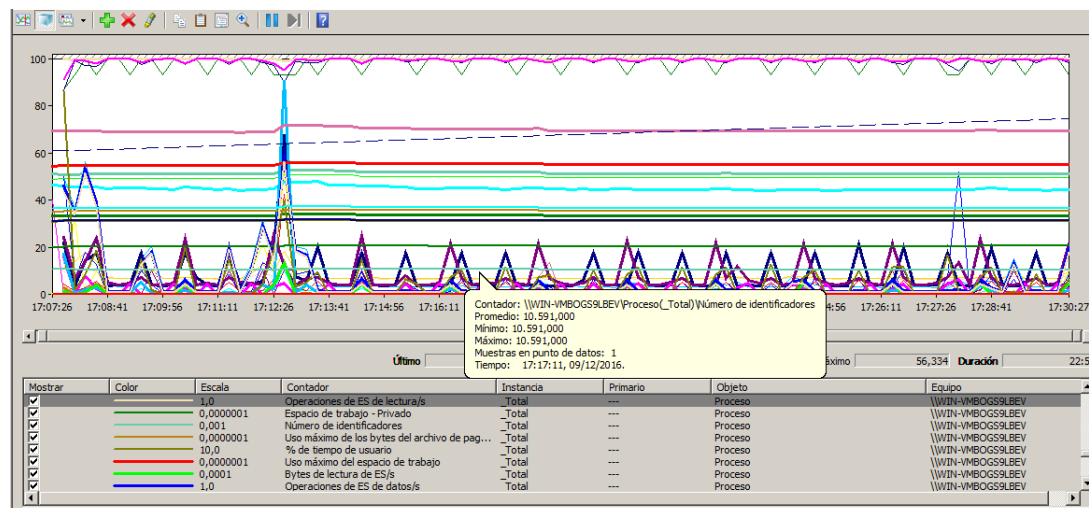


Figura 0.23: Resultado del recopilador de datos

Si pinchamos en algún identificador, como en 0.24 con las interrupciones, se nos muestra el último dato, el promedio, el mínimo, y el máximo.

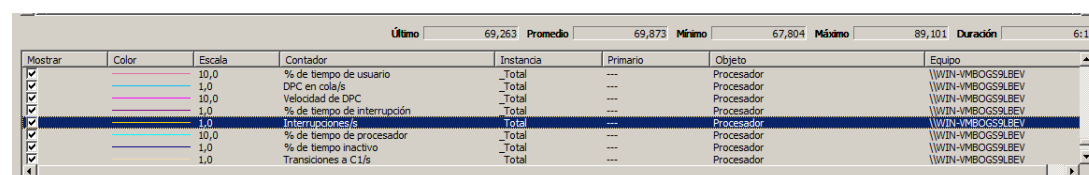


Figura 0.24: Identificadores del gráfico

En 0.25, podemos comprobar el porcentaje del tiempo que se ha ejecutado el procesador en modo privilegiado. En este caso ha sido casi del 100%, el último dato registrado ha sido de 99,479 % y el promedio de 99,571 %.

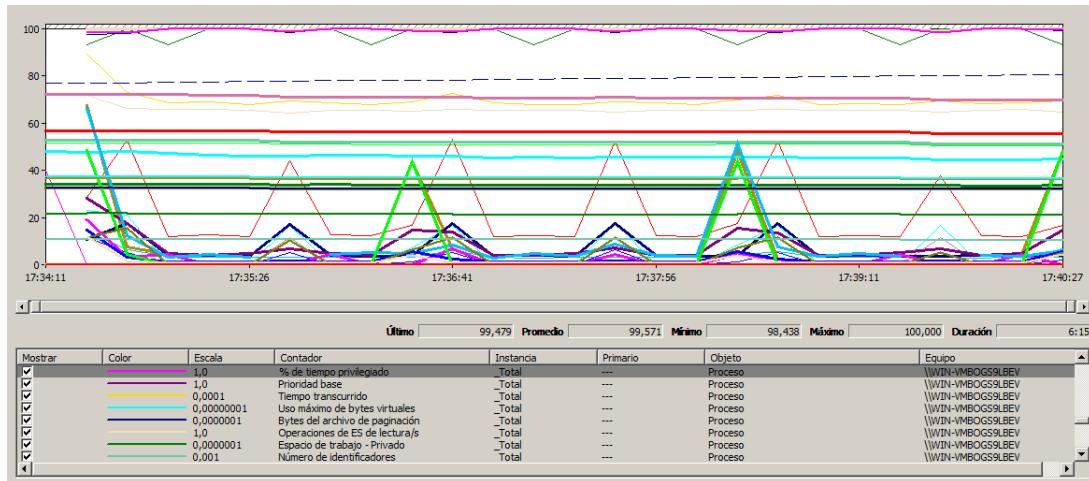


Figura 0.25: Porcentaje de tiempo en modo privilegiado

Cuestión 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. Monitoree varios parámetros y haga capturas de pantalla de lo que está mostrando comentando qué observa.

La demo de **Munin** se encuentra en [6]. En 0.26 se muestra la vista general de **Munin**, donde se pueden ver algunas de las funcionalidades de las que dispone esta herramienta. Podemos clasificar las estadísticas recogidas por día, semana, mes o año. En mi caso he utilizado los datos recopilados por año.

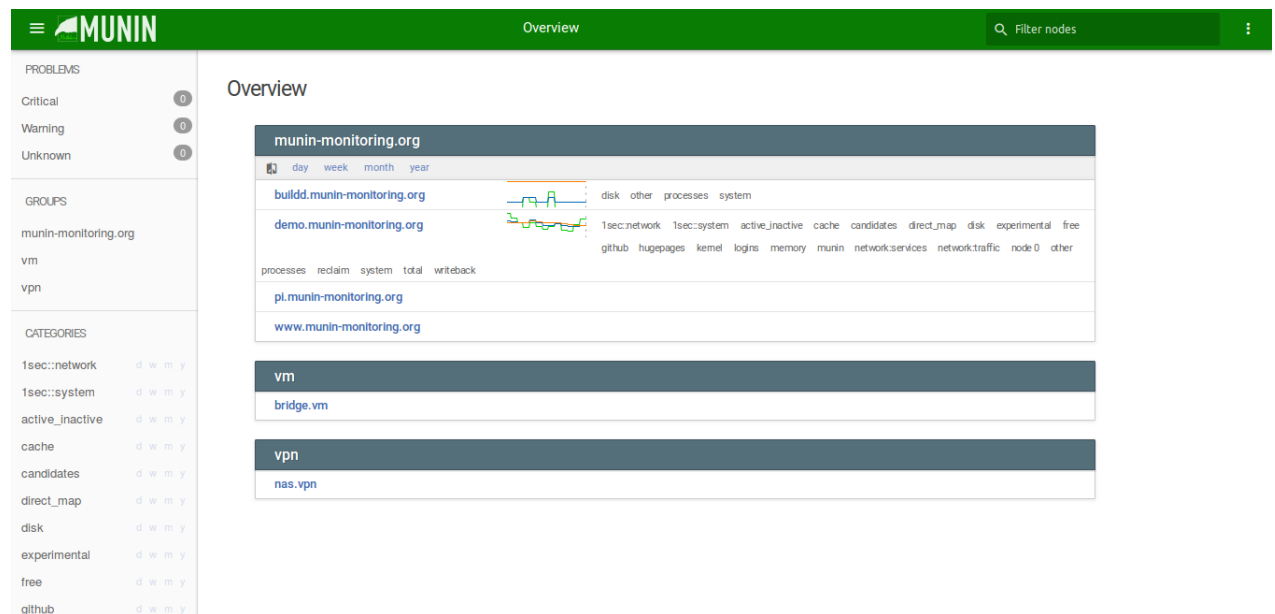


Figura 0.26: Vista general de **Munin**

Entre las estadísticas que podemos consultar, como se muestra en 0.27, se encuentran las referentes a la interfaz de red, al uso del disco, a la memoria, a los procesos, o al sistema.

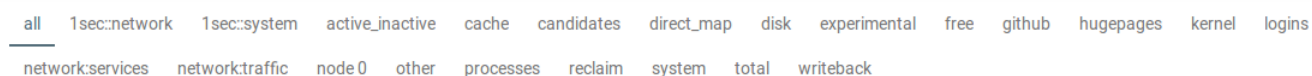


Figura 0.27: Gráficos de **Munin**, resultados por año

Como se muestra en 0.28, en **Munin** se han utilizado como máximo 97 hebras en el último año, como promedio 70.

Dynazoom - Number of threads

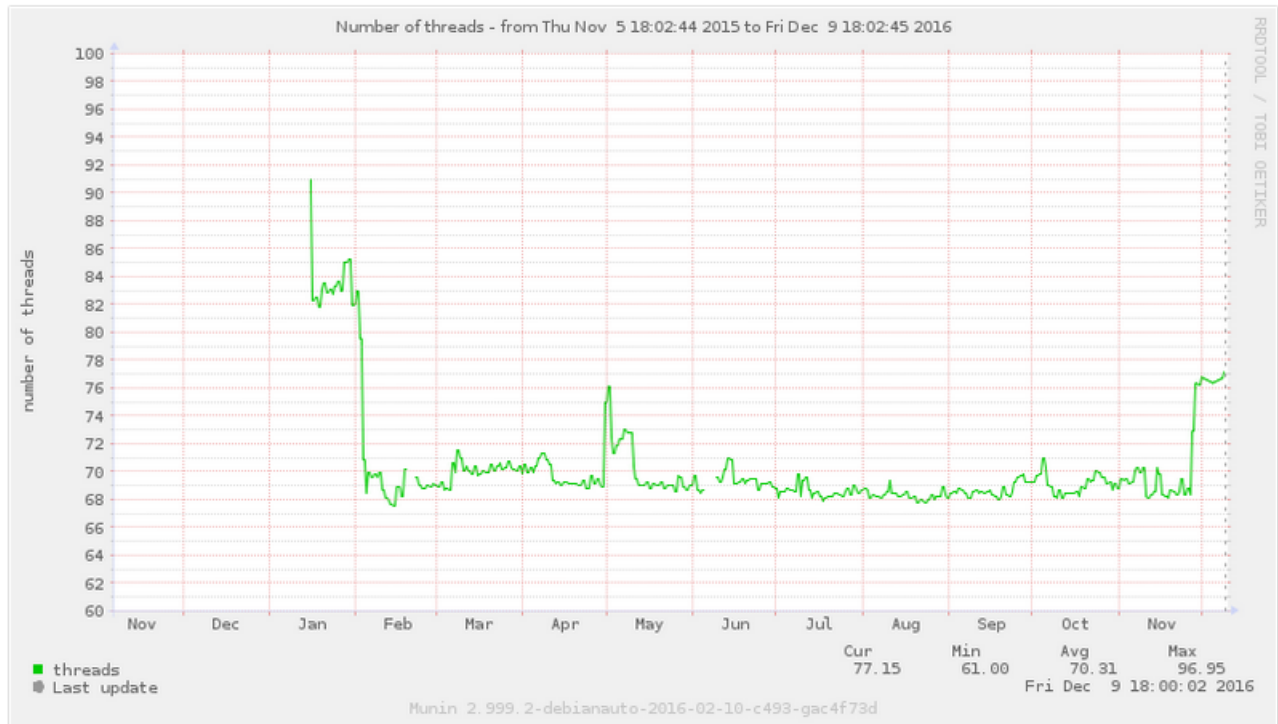


Figura 0.28: Gráfico sobre el número de hebras utilizadas

En 0.29 se muestran las interrupciones dadas en el último año, así como los cambios de contexto que se han producido.

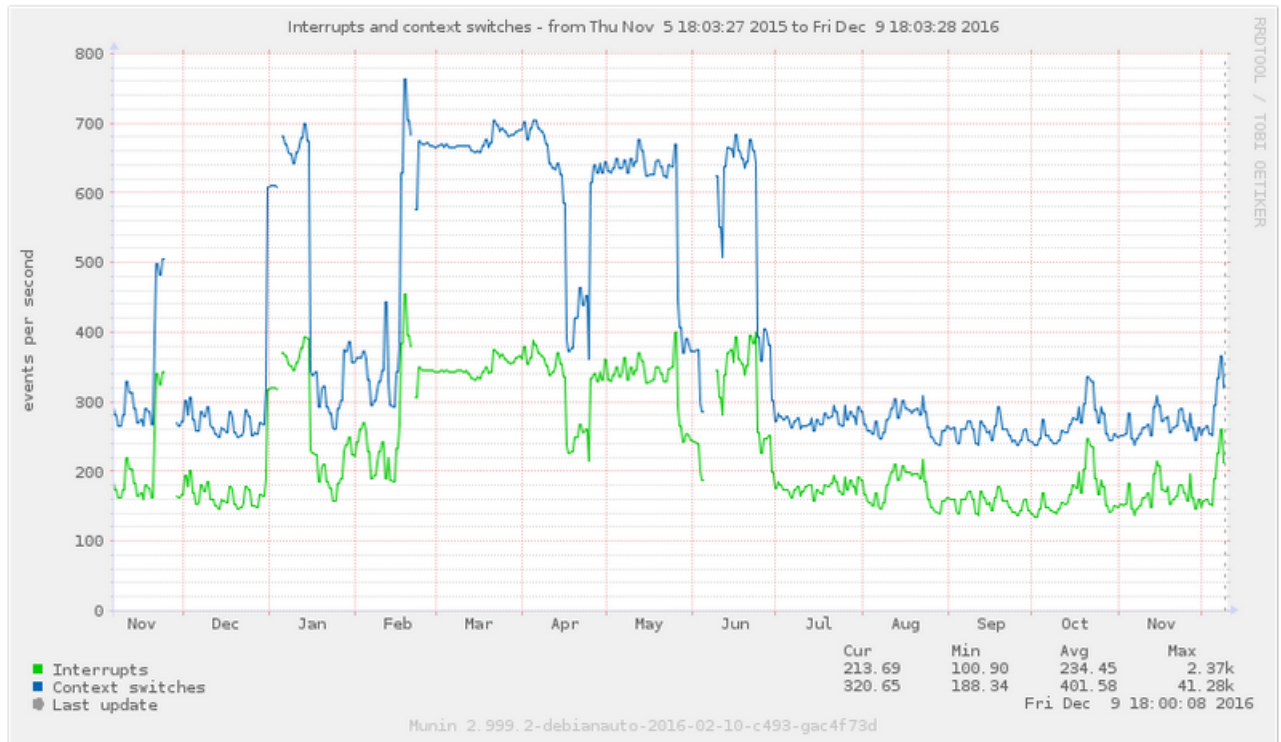


Figura 0.29: Gráfico sobre interrupciones del sistema

En 0.30 se muestra un gráfico detallado del uso de la memoria en el servidor. Entre otros datos se detallan el número de paginaciones, el *swap*, memoria no utilizada, memoria *mapeada*, activa e inactiva.

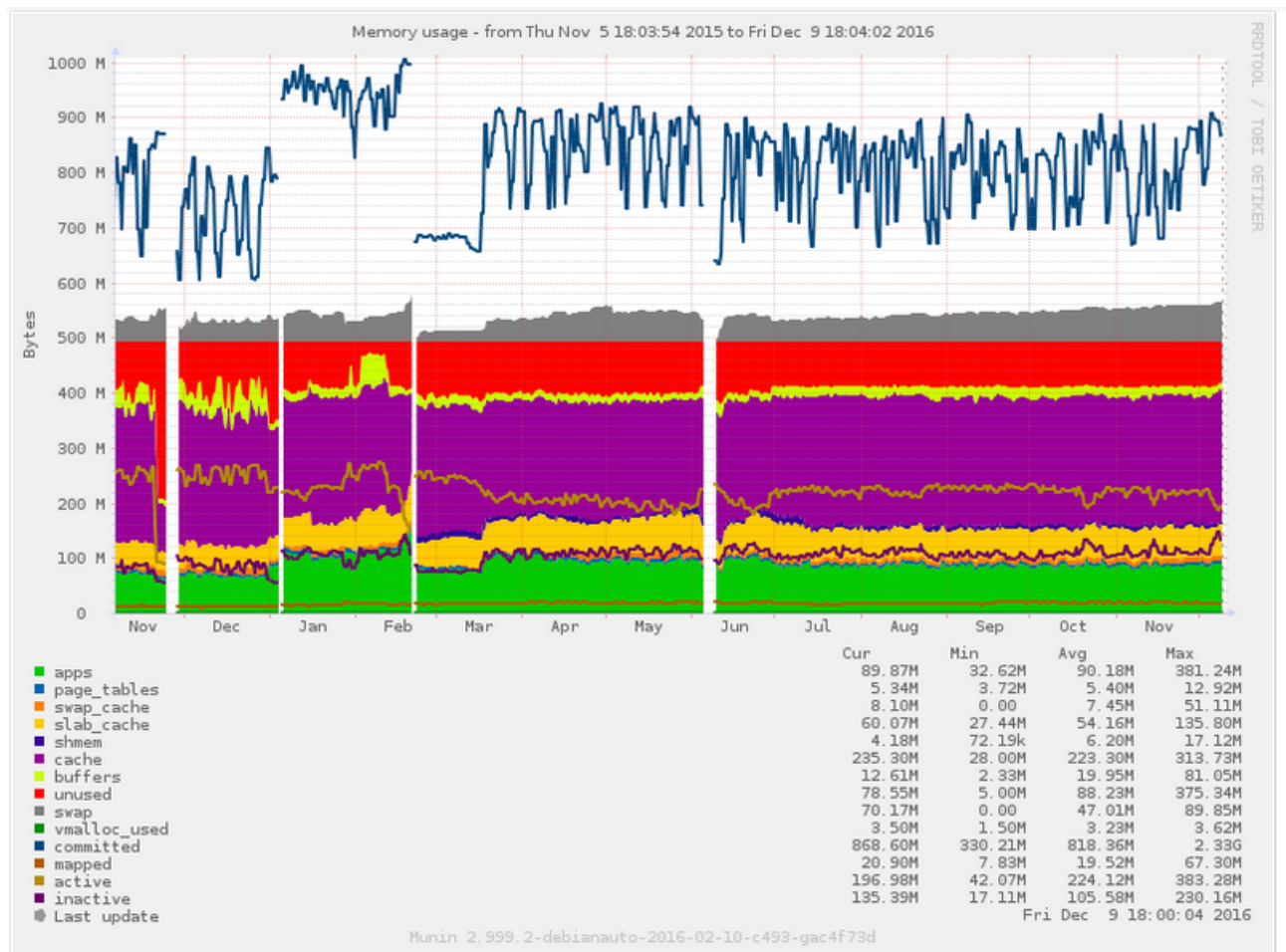


Figura 0.30: Gráfico sobre memoria utilizada

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

He escogido el artículo que se encuentra en [13]. En este, se habla de que son muchas las ocasiones en las que un administrador de sistema se ve en una situación en la que no sabe cómo ha ocurrido un determinado error, o un evento inesperado en el sistema, y no existe ningún fichero *log* donde se expliquen dichos errores. Para estos casos, la utilización de la herramienta **strace** es realmente potente para monitorizar llamadas al sistema y señales. Debe ser ejecutado con permisos de *root* para aprovechar toda su funcionalidad.

En el artículo se nos describen casos concretos en los que **strace** tiene gran utilidad. Por ejemplo, cuando no conocemos el fichero de *logs* de una aplicación, sencillamente

podemos seguir su traza y filtrar la misma con una concatenación de *grep log*. De esta forma será muy fácil saber cuáles son los ficheros de *log* de ese programa.

También podemos encontrar errores en la ejecución de un programa filtrando la traza del mismo con *grep -1*, ya que esta es la forma que **strace** tiene de decirnos que en la ejecución de esa línea ha ocurrido un error.

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

El script que he utilizado se presenta a continuación (y será añadido al archivo comprimido de la entrega como *prof.py*):

```
#!/usr/bin/env python3
import profile

def fib(n):
    if n == 0:
        return 0

    elif n == 1:
        return 1

    else:
        return fib(n-1)+fib(n-2)

def loop():
    n = 0
    while n < 100000000:
        n = n+1

def main():
    print ("IN LOOP")
    loop()
    print ("FIB(30)")
    fib(30)

profile.run('main()')
```

El código al que vamos a pasar el *profiler* se resume en dos funciones en **Python**, una de ellas calcula el término *n* de la sucesión de **Fibonacci**, y la otra es un bucle desde 0 hasta 10 millones dentro del cual se incrementa una variable.

Para analizar el comportamiento del mismo con el *profiler* de **Python**, he recurrido a la biblioteca *profile*, que se explica en [12].

Como podemos ver en 0.31, el bucle tarda 2.096 segundos en ejecutarse mientras que la función de **Fibonacci** tarda 13.593 segundos.

```
alvarogl dec 2016 > ./prof.py
IN LOOP
FIB(30)
El término fib(30) es 832040
      2692546 function calls (10 primitive calls) in 15.691 seconds

Ordered by: standard name

ncalls  tottime  percall  cumtime  percall filename:lineno(function)
      1   0.000    0.000   15.690    15.690 :0(exec)
      3   0.000    0.000    0.000     0.000 :0(print)
      1   0.002    0.002    0.002     0.002 :0(setprofile)
      1   0.000    0.000   15.689    15.689 <string>:1(<module>)
      1   2.096    2.096    2.096     2.096 prof.py:14(loop)
      1   0.000    0.000   15.689    15.689 prof.py:19(main)
2692537/1 13.593    0.000   13.593    13.593 prof.py:4(fib)
      1   0.000    0.000   15.691    15.691 profile:0(main())
      0   0.000    0.000    0.000     0.000 profile:0(profiler)
```

Figura 0.31: Resultado del *profile* de la primera versión

Habría posibilidad de mejorar este código, ya que la función que más tiempo de ejecución consume es la de **Fibonacci**, pero existen otros algoritmos más eficientes para hacer este cálculo.

El script que he utilizado para mejorar el código se presenta a continuación (y será añadido al archivo comprimido de la entrega como *prof_mejora.py*):

```
#!/usr/bin/env python3
import profile

def fib(n):
    a, b = 0, 1
    for _ in range(n):
        a, b = b, a+b
    return a

def loop():
    n = 0
    while n < 100000000:
        n = n+1

def main():
    print ("IN LOOP")
    loop()
    print ("FIB(30)")
    fib(30)
```

```
profile.run('main()')
```

En este caso he utilizado una versión iterativa del algoritmo para calcular un término n de **Fibonacci**, y el resultado, como se muestra en 0.32, es notablemente mejor, pues el tiempo de ejecución de **Fibonacci** es inferior a 1 milisegundo (el *profiler* indica que el tiempo es de 0.000 segundos).

```
alvarogl dec 2016 > ./prof_mejora.py
IN LOOP
FIB(30)
El término fib(30) es 832040
10 function calls in 1.849 seconds

Ordered by: standard name

ncalls  tottime  percall  cumtime  percall filename:lineno(function)
1      0.000    0.000    1.848    1.848  :0(exec)
3      0.000    0.000    0.000    0.000  :0(print)
1      0.001    0.001    0.001    0.001  :0(setprofile)
1      0.000    0.000    1.848    1.848  <string>:1(<module>)
1      1.847    1.847    1.847    1.847  prof_mejora.py:10(loop)
1      0.000    0.000    1.848    1.848  prof_mejora.py:15(main)
1      0.000    0.000    0.000    0.000  prof_mejora.py:4(fib)
1      0.000    0.000    1.849    1.849  profile:0(main())
0      0.000    0.000    0.000    0.000  profile:0(profiler)
```

Figura 0.32: Resultado del *profile* de la segunda versión

Cuestión 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)

En mi caso, he buscado una base de datos de ejemplo de la página oficial de **MySQL** que se puede encontrar en [10]. En concreto, he escogido la de *world*.

Se han consultado las referencias de [8] y [9] para la realización del ejercicio.

En primer lugar, es necesario acceder a la base de datos, para ello, como se explica en [7], ejecutamos:

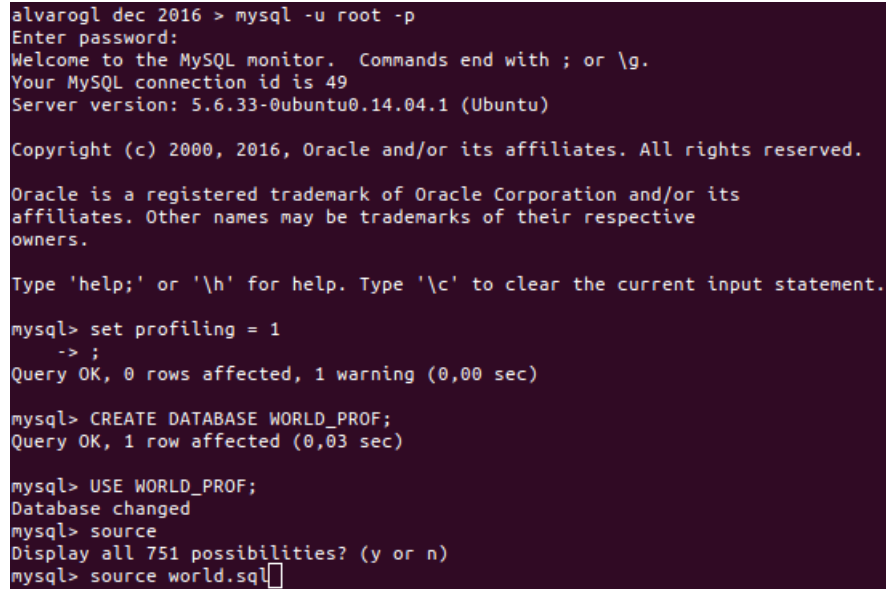
```
mysql -uroot -p
```

Para activar el *profiling* se utiliza el *mysql-command*:

```
set profiling = 1;
```

Creamos la base de datos, la utilizamos e importamos *world.sql*, que fue descargado de [10] (dicho archivo será adjuntado al comprimido de la entrega). Lo hacemos de esta manera, como se ve en 0.33:

```
create database world;
use world;
source world.sql
```

A screenshot of a terminal window with a dark background and light-colored text. The terminal shows a MySQL command-line interface session. The user connects as 'root' and enters a password. The MySQL version is 5.6.33-0ubuntu0.14.04.1. The user sets profiling to 1, creates a database named 'WORLD_PROF', switches to it, and then sources a file named 'world.sql'. The terminal output includes standard MySQL startup messages and command execution results.

```
alvarogl dec 2016 > mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 49
Server version: 5.6.33-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> set profiling = 1
-> ;
Query OK, 0 rows affected, 1 warning (0,00 sec)

mysql> CREATE DATABASE WORLD_PROF;
Query OK, 1 row affected (0,03 sec)

mysql> USE WORLD_PROF;
Database changed
mysql> source
Display all 751 possibilities? (y or n)
mysql> source world.sql
```

Figura 0.33: Creación y configuración del *profiler*

Para comprobar los resultados del *profile*, simplemente ejecutamos **SHOW PROFILES** para ver datos generales de la ejecución, como se aprecia en 0.34, y **SHOW PROFILE** (ver 0.35) para incidir sobre una *Query* en concreto.

```
mysql> mysql> SHOW PROFILES;
```

Query_ID	Duration	Query
5336	0.00011675	INSERT INTO `countrylanguage` VALUES ('ZMB','Tongan','F',11.0)
5337	0.00011675	INSERT INTO `countrylanguage` VALUES ('ZWE','English','T',2.2)
5338	0.00012825	INSERT INTO `countrylanguage` VALUES ('ZWE','Ndebele','F',16.2)
5339	0.00011900	INSERT INTO `countrylanguage` VALUES ('ZWE','Nyanja','F',2.2)
5340	0.00011725	INSERT INTO `countrylanguage` VALUES ('ZWE','Shona','F',72.1)
5341	0.04852525	COMMIT
5342	0.00010200	SET AUTOCOMMIT=1
5343	0.00008500	/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */
5344	0.00006875	/*!40101 SET SQL_MODE=@OLD_SQL_MODE */
5345	0.00006625	/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */
5346	0.00005625	/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */
5347	0.00007125	/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */
5348	0.00006225	/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */
5349	0.00006050	/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */
5350	0.00005575	/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */

```
15 rows in set, 1 warning (0,00 sec)
```

Figura 0.34: Ejecución de **SHOW PROFILES**

```
mysql> SHOW PROFILE FOR QUERY 5341;
```

Status	Duration
starting	0.048439
query end	0.000019
closing tables	0.000009
freeing items	0.000036
cleaning up	0.000023

```
5 rows in set, 1 warning (0,00 sec)
```

Figura 0.35: Ejecución de **SHOW PROFILE**

He ejecutado dos ejemplos más para hacer un *profiling* de la creación de una base de datos, en 0.36 y otro de una consulta, en 0.37.

```
mysql> CREATE TABLE MyGuests ( id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY, firstname VARCHAR(30) NOT NULL, lastname VARCHAR(30) NOT NULL, email VARCHAR(50), reg_date TIMEST );
Query OK, 0 rows affected (0,28 sec)

mysql> SHOW PROFILES;
```

Query_ID	Duration	Query
1	0.00069325	CREATE TABLE MyGuests (id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY, firstname VARCHAR(30) NOT NULL, lastname VARCHAR(30) NOT NULL, email VARCHAR(50), reg_date TIMEST
2	0.00036300	create database ola
3	0.00019275	SELECT DATABASE()
4	0.00058000	show databases
5	0.00031700	show tables
6	0.27271025	CREATE TABLE MyGuests (id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY, firstname VARCHAR(30) NOT NULL, lastname VARCHAR(30) NOT NULL, email VARCHAR(50), reg_date TIMEST

```
6 rows in set, 1 warning (0,00 sec)
```

Figura 0.36: *Profile* de la creación de una base de datos

```
mysql> show profiles;
```

Query_ID	Duration	Query
21392	0.00014550	INSERT INTO `countrylanguage` VALUES ('ZWE','Ndebele','F',16.2)
21393	0.00014675	INSERT INTO `countrylanguage` VALUES ('ZWE','Nyanja','F',2.2)
21394	0.00014600	INSERT INTO `countrylanguage` VALUES ('ZWE','Shona','F',72.1)
21395	0.05824500	COMMIT
21396	0.00010450	SET AUTOCOMMIT=1
21397	0.00008800	/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */
21398	0.00007675	/*!40101 SET SQL_MODE=@OLD_SQL_MODE */
21399	0.00007550	/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */
21400	0.00006725	/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */
21401	0.00008400	/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */
21402	0.00007950	/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */
21403	0.00007000	/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */
21404	0.00006300	/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */
21405	0.00017050	set profiling = 1
21406	0.01506200	select * from city

15 rows in set, 1 warning (0.00 sec)

```
mysql> show profile for query 21406;
```

Status	Duration
starting	0.000077
checking permissions	0.000014
Opening tables	0.000029
init	0.000028
System lock	0.000011
optimizing	0.000008
statistics	0.000018
preparing	0.000017
executing	0.000004
Sending data	0.014769
end	0.000018
query end	0.000011
closing tables	0.000019
freeing items	0.000023
cleaning up	0.000019

15 rows in set, 1 warning (0.00 sec)

Figura 0.37: *Profile* de una consulta

Referencias

- [1] *Documentación oficial de Debian, Debian package management*
<https://www.debian.org/doc/manuals/debian-reference/ch02.en.html>.
- [2] *Documentación oficial de Fedora, Capítulo 4. Yum*
https://docs.fedoraproject.org/en-US/Fedora/15/html/Deployment_Guide/ch-yum.html.
- [3] *Linux man page, cron*
<https://linux.die.net/man/5/crontab>.
- [4] *Linux man page, date*
<https://linux.die.net/man/1/date>.
- [5] *Linux man page, gunzip*
<https://linux.die.net/man/1/gunzip>.
- [6] *Munin*
<http://demo.munin-monitoring.org/munin-monitoring.org/demo.munin-monitoring.org/>.
- [7] *MySQL documentation, command options*
<http://dev.mysql.com/doc/refman/5.7/en/mysql-command-options.html>.
- [8] *MySQL documentation, profile*
<http://dev.mysql.com/doc/refman/5.5/en/show-profile.html>.
- [9] *MySQL documentation, profiles*
<http://dev.mysql.com/doc/refman/5.7/en/show-profiles.html>.
- [10] *MySQL documentation*
<http://dev.mysql.com/doc/index-other.html>.
- [11] *Programación de tareas repetitivas del sistema (cron)*
https://docs.oracle.com/cd/E24842_01/html/E23086/sysrescron-1.html.
- [12] *Python official documentation*
<https://docs.python.org/2/library/profile.html>.
- [13] *Sysadmin tips*
http://blog.softlayer.com/2013/sysadmin-tips-and-tricks-using-strace-to-monitor-system-utm_source=twitter&utm_medium=social&utm_content=beyond-the-command-line-with-strace&utm_campaign=blog-development-tips-and-tricks.
- [14] *Ubuntu man page, dmesg*
<http://manpages.ubuntu.com/manpages/wily/man1/dmesg.1.html>.

- [15] Helmke Matthew. *Ubuntu Unleashed 2015 Edition: Covering 14.10 and 15.04*. Sams Publishing, 2014.
- [16] Przemysław Skibinski and Jakub Swacha. Fast and efficient log file compression, <http://ceur-ws.org/Vol-325/paper06.pdf>. 2007.