

INGENIERÍA DE SERVIDORES: PRÁCTICA 3

Monitorización de servicios

Montserrat Rodríguez Zamorano

30 de mayo de 2016

Índice

1.	1
1.1. ¿Qué archivo le permite ver qué programas se han instalado con el gestor de paquetes?	1
1.2. ¿Qué significan las terminaciones .1.gz o .2.gz de los archivos en ese directorio? . .	1
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.	2
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.	3
4. Ejecute el monitor de “System Performance” y muestre el resultado. Incluya capturas de pantalla comentando la información que aparece.	4
5. Cree un recopilador de datos definido por el usuario (modo avanzado) que incluya tanto el contador de rendimiento como los datos de seguimiento.	6
6. Instale alguno de los monitores comentados arriba en su máquina y pruebe a ejecutarlos (tenga en cuenta que si lo hace en la máquina virtual, los resultados pueden no ser realistas). Alternativamente, busque otros monitores para hardware comerciales o de código abierto para Windows y Linux.	9
7. 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.	10
8. Escriba un breve resumen sobre alguno de los artículos donde se muestra el uso de strace, o busque otro programa similar y coméntelo.	11
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).	12
10. Cuestiones opcionales	14
10.1. Cuestión opcional 2. Instale Nagios en su sistema documentando el proceso y muestre el resultado de la monitorización de su sistema comentando qué aparece.	14
10.2. Cuestión opcional 3. Haga lo mismo que con Munin.	18
10.3. Cuestión opcional 5: Pruebe a instalar este monitor en alguno de sus tres sistemas. Realice capturas de pantalla del proceso de instalación y comente capturas de pantalla del proceso en ejecución.	19
11. Capturas de pantalla	22
11.1. Cuestión 5	22

Índice de figuras

1. Fichero /var/log/apt/history.log	1
2. Fichero /var/log/apt/term.log	1
3. Listado de crontabs programados para el usuario y para root	2
4. Programando una tarea diaria con crontab (1)	3
5. Programando una tarea diaria con crontab (2)	3
6. Salida por pantalla previa a la conexión del dispositivo USB	4
7. Salida por pantalla tras la conexión del dispositivo USB	4

8.	Salida por pantalla tras la desconexión del dispositivo USB	4
9.	Herramientas de supervisión: monitor de rendimiento	5
10.	Recopiladores de datos del sistema	5
11.	Información general de recursos	5
12.	Información acerca de la CPU: sistema	6
13.	Informe de diagnóstico I	7
14.	Datos de seguimiento referentes al servicio web	7
15.	Datos de seguimiento referentes al proceso	8
16.	Informe de diagnóstico II	8
17.	Consultando la temperatura del dispositivo <code>sda</code>	9
18.	Interfaz gráfica de <code>xsensors</code>	9
19.	Interfaz gráfica de <code>psensor</code> [9]	10
20.	Monitorizando los recursos de un servidor: procesos de <code>apache</code>	10
21.	Monitorizando los recursos de un servidor: consultas de <code>mysql</code>	11
22.	Monitorizando los recursos de un servidor: procesos	11
23.	Creación de la base de datos <code>prueba</code>	12
24.	Creación de la tabla <code>tablaprueba</code>	12
25.	Añadiendo datos a la tabla <code>tablaprueba</code>	12
26.	Ejecución de <code>show profile</code>	13
27.	Profile de la consulta 14	13
28.	Accediendo a <code>Nagios</code> desde CentOS	15
29.	Instalando <code>Nagios</code> en Ubuntu Server I	16
30.	Instalando <code>Nagios</code> en Ubuntu Server II	16
31.	Accediendo a <code>Nagios</code> desde Ubuntu Server	16
32.	Monitorizando con <code>Ganglia</code> I	16
33.	Monitorizando con <code>Ganglia</code> II	17
34.	Monitorizando con <code>Ganglia</code> III	17
35.	Monitorizando con <code>Ganglia</code> IV	17
36.	Monitorizando con <code>Ganglia</code>	18
37.	Monitorizando con <code>Ganglia</code> : uso de memoria	18
38.	Monitorizando con <code>Ganglia</code> : uso de la CPU	19
39.	Instalación de <code>Cacti</code> I	19
40.	Instalación de <code>Cacti</code> II	19
41.	Autenticación en <code>Cacti</code>	20
42.	Accediendo a <code>Cacti</code>	20
43.	Monitorizando con <code>Cacti</code> : uso de la memoria y carga	21
44.	Monitorización con <code>Cacti</code> : procesos	21
45.	Creación de un recopilador de datos definido por el usuario (modo avanzado)	22
46.	Incluir el contador de rendimiento y los datos de seguimiento	22
47.	Datos de seguimiento: procesador, proceso, servicio web	23
48.	Almacenar el resultado en el directorio Escritorio/logs	23

1.

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

La información acerca de las acciones que se han realizado con el gestor de paquetes `apt` puede buscarse en el directorio `/var/log/apt`. En esta carpeta pueden encontrarse dos archivos: `history.log` y `term.log`. El archivo que permite consultar los programas que se han instalado es `history.log`, que permite listar la historia más reciente [1]. En la captura de pantalla (1) se puede ver parte del contenido del archivo `/var/log/apt/history.log`. Como se puede observar, en el archivo aparece la línea de comandos que se utilizó y la fecha y hora exactas de inicio y final de la instalación.

```
Start-Date: 2016-05-14  00:12:14
Commandline: apt-get install mysql-server php5-mysql
Install: php5-mysql:amd64 (5.5.9+dfsg-1ubuntu4.16)
Remove: php5-mysqld:amd64 (5.5.9+dfsg-1ubuntu4.16)
End-Date: 2016-05-14  00:12:16
```

Figura 1: Fichero `/var/log/apt/history.log`

Por otra parte, el archivo `/var/log/apt/term.log` se crea para complementar al primer archivo y contiene las salidas por pantalla que se producen en la terminal durante el uso de `apt`. Esto es útil, especialmente en caso de error, ya que puede consultarse lo que ha ocurrido durante la instalación de un programa [2]. En la captura de pantalla (2) se puede ver parte del contenido del archivo `/var/log/apt/term.log`.

```
Log started: 2016-05-13  11:50:34
Seleccionando el paquete php5-mysql previamente no seleccionado.
(Leyendo la base de datos ... 166338 ficheros o directorios instalados actualmente.)
Preparing to unpack .../php5-mysql_5.5.9+dfsg-1ubuntu4.16_amd64.deb ...
Unpacking php5-mysql (5.5.9+dfsg-1ubuntu4.16) ...
Seleccionando el paquete mysql-server previamente no seleccionado.
Preparing to unpack .../mysql-server_5.5.49-0ubuntu0.14.04.1_all.deb ...
Unpacking mysql-server (5.5.49-0ubuntu0.14.04.1) ...
Configurando php5-mysql (5.5.9+dfsg-1ubuntu4.16) ...
Configurando mysql-server (5.5.49-0ubuntu0.14.04.1) ...
Log ended: 2016-05-13  11:50:36
```

Figura 2: Fichero `/var/log/apt/term.log`

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

Con el uso del gestor de paquetes, es intuitivo que el archivo `history.log` se hará cada vez más grande. Cuando el tamaño de un archivo es demasiado grande, Linux, de forma interna, sustituye este archivo por uno comprimido con la extensión `.gz`, con el mismo contenido, permisos,... y los numera en los ficheros `.1.gz`, `.2.gz` y así sucesivamente. Es importante tener en cuenta que la numeración del fichero será más alta cuando éste sea más antiguo, de forma que cuando se genere un nuevo archivo comprimido, será necesario renombrar todos los anteriores. El comando `zcat` muestra el contenido del archivo por pantalla sin tener que descomprimirlo [3]:

```
zcat -f /var/log/apt/history.log.1.gz
```

donde la opción `-f` se utiliza para forzar la compresión o descompresión [3].

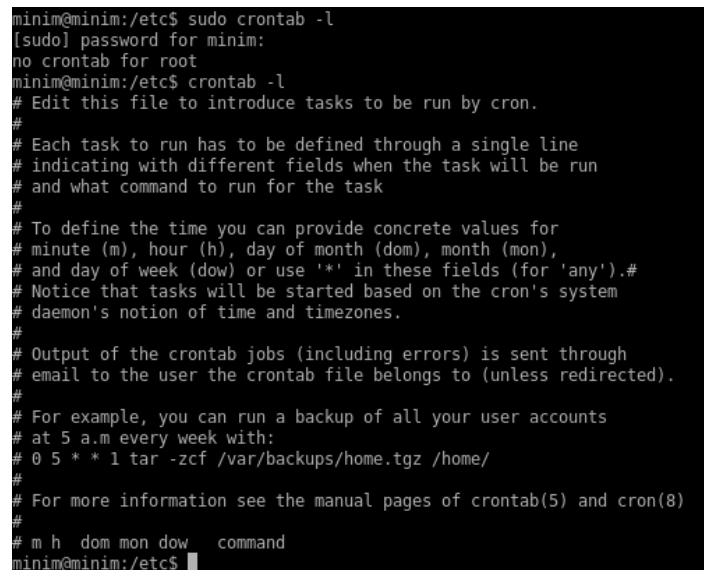
2. ¿Qué archivo ha de modificar para programar una tarea? Escriba la línea necesaria para ejecutar una vez al día una copia del directorio /codigo a /seguridad/\$fecha donde \$fecha es la fecha actual.

El comando `crontab` se utiliza para automatizar y programar tareas. El archivo del mismo nombre contiene las instrucciones del tipo *Ejecuta lo siguiente cada x tiempo* [4]. Para ver las tareas que están programadas y editarlas se ejecutan, respectivamente, los siguientes comandos:

```
crontab -l
```

```
crontab -e
```

Es importante tener en cuenta que el resultado no será el mismo si se ejecutan estos comandos como superusuario, como puede verse en la figura (3).



```
minim@minim:/etc$ sudo crontab -l
[sudo] password for minim:
no crontab for root
minim@minim:/etc$ crontab -l
# 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
minim@minim:/etc$
```

Figura 3: Listado de `crontabs` programados para el usuario y para `root`

Probamos ahora a programar la tarea. Para ello, es necesario tener en cuenta la sintaxis que sigue. Si ejecutamos el comando correspondiente para editar tareas, podemos ver como dicha sintaxis está explicada al inicio del documento.

Para incluir una tarea, tendremos que escribir en el siguiente orden: minutos, horas, día del mes, mes, día de la semana. Si se deja un asterisco en alguna columna, significará que queremos que la tarea se lleve a cabo en cada minuto, hora,... dependiendo de en qué columna esté dicho asterisco [5].

Añadimos la tarea en el archivo tal y como puede verse en la captura de pantalla (4). Como queremos que se ejecute sólo una vez al día, se especificará el minuto y la hora, cada día del mes (*), cada mes (*), cada día de la semana (*). El directorio `codigo` contiene tres archivos llamados `codigo1.txt`, `codigo2.txt` y `codigo3.txt` no vacíos, y `script_seguridad.sh` es un script con el siguiente contenido.

```
date_time=$(date '+%y%m%d_%H%M%S')
mkdir -p ~/seguridad/$date\_time
cp ~/seguridad/$date\_time
```

Es decir, se creará una carpeta con la fecha y hora de la copia de seguridad, dentro de la carpeta `seguridad`, en la que se copiará todo el contenido del directorio `codigo`. Una vez haya pasado la hora

```
GNU nano 2.2.6 Archivo: /tmp/crontab.h43Dut/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
40 14 * * * ~/script_seguridad.sh
```

Figura 4: Programando una tarea diaria con `crontab` (1)

a la que se ha programado la tarea, comprobamos que se ha hecho la copia correctamente (5).

```
minim@minim:~/seguridad$ crontab -e
crontab: installing new crontab
minim@minim:~/seguridad$ ls
160529_144001
```

Figura 5: Programando una tarea diaria con `crontab` (2)

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.

Nota: en lugar de copiar y pegar la salida del comando se ha hecho una captura de pantalla ya que, al tratarse de una máquina virtual, era bastante complicado pegar la salida en este documento, que se encontraba en otra máquina.

Tenemos en cuenta que el comando `dmesg` se utiliza para mostrar por pantalla una lista con mensajes del kernel [6] y que la salida por pantalla es de la siguiente forma `[time] device name : message`, donde `[time]` es el número de segundos que han pasado desde el arranque [7]. En primer lugar se utiliza `dmesg | tail` para obtener la información previa a la conexión del dispositivo (6).

Una vez hecho esto, conectamos el dispositivo USB y observamos la salida por pantalla (7). Esta vez los mensajes muestran información acerca del dispositivo. Por ejemplo: en las primeras líneas podemos ver el tipo dispositivo que se ha introducido, **Generic Flash Disk**; que no está protegido para la escritura en **Write Protect is off** o que el dispositivo se ha montado en **sdcl**.

Probamos a desconectar el dispositivo (8). Esta vez, la salida muestra un “cambio de capacidad” a 0, en el mensaje enviado en el segundo 326.725855, para posteriormente detectar la desconexión del USB en el siguiente mensaje.

```

minim@minim:~$ dmesg | tail
[ 238.531184] init: plymouth-upstart-bridge main process ended, respawning
[ 241.021394] init: anacron main process (2004) killed by TERM signal
[ 245.718124] ip tables: (C) 2000-2006 Netfilter Core Team
[ 248.288911] init: plymouth-stop pre-start process (2951) terminated with status 1
[ 260.076491] audit: printk_skb: 84 callbacks suppressed
[ 260.076492] audit: type=1400 audit(1464447514.275:46): apparmor="DENIED" operation="open" profile="/usr/lib/telepathy/mission-control-5" name="/home/minim/.config/dconf/user" pid=3432 comm="mission-control-5" requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
[ 265.643160] audit: type=1400 audit(1464447518.940:47): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/lib/cups/backend/cups-pdf" pid=3489 comm="apparmor_parser"
[ 265.643169] audit: type=1400 audit(1464447518.940:48): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=3489 comm="apparmor_parser"
[ 265.643533] audit: type=1400 audit(1464447518.940:49): apparmor="STATUS" operation="profile_replace" profile="unconfined" name="/usr/sbin/cupsd" pid=3489 comm="apparmor_parser"
[ 322.969153] valid eCryptfs headers not found in file header region or xattr region, inode 34712
minim@minim:~$

```

Figura 6: Salida por pantalla previa a la conexión del dispositivo USB

```

minim@minim:~$ dmesg | tail
[ 231.922444] usbcore: registered new interface driver usb-storage
[ 231.927257] usbcore: registered new interface driver uas
[ 232.950721] scsi 4:0:0:0: Direct-Access Generic Flash Disk 8.00 PQ: 0 ANSI: 4
[ 232.951096] sd 4:0:0:0: Attached scsi generic sg3 type 0
[ 232.986343] sd 4:0:0:0: [sdc] 15769600 512-byte logical blocks: (8.07 GB/7.51 GiB)
[ 232.997153] sd 4:0:0:0: [sdc] Write Protect is off
[ 232.997162] sd 4:0:0:0: [sdc] Mode Sense: 23 00 00 00
[ 233.008037] sd 4:0:0:0: [sdc] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[ 233.085112] sdc: sdcl
[ 233.147307] sd 4:0:0:0: [sdc] Attached SCSI removable disk
minim@minim:~$

```

Figura 7: Salida por pantalla tras la conexión del dispositivo USB

```

minim@minim:~$ dmesg | tail
[ 232.950721] scsi 4:0:0:0: Direct-Access Generic Flash Disk 8.00 PQ: 0 ANSI: 4
[ 232.951096] sd 4:0:0:0: Attached scsi generic sg3 type 0
[ 232.986343] sd 4:0:0:0: [sdc] 15769600 512-byte logical blocks: (8.07 GB/7.51 GiB)
[ 232.997153] sd 4:0:0:0: [sdc] Write Protect is off
[ 232.997162] sd 4:0:0:0: [sdc] Mode Sense: 23 00 00 00
[ 233.008037] sd 4:0:0:0: [sdc] Write cache: disabled, read cache: enabled, doesn't support DPO or FUA
[ 233.085112] sdc: sdcl
[ 233.147307] sd 4:0:0:0: [sdc] Attached SCSI removable disk
[ 326.725855] sdc: detected capacity change from 8074035200 to 0
[ 331.010136] usb 1-2: USB disconnect, device number 3
minim@minim:~$

```

Figura 8: Salida por pantalla tras la desconexión del dispositivo USB

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

Para iniciar **perfmon**, se ejecuta el comando **perfmon** desde la consola. Una vez hecho esto, se abre el monitor de rendimiento y podemos observar, entre otras cosas, herramientas de supervisión (ver (9)) o recopiladores de datos del sistema (ver (10)).

Para que se produjesen los picos en la imagen (9), se han abierto varios programas a la vez, para observar los cambios en la gráfica. Esto puede resultar muy útil, por ejemplo, para detectar una situación de sobrecarga.

- En *Recopiladores de datos del sistema*, seleccionamos *System Performance*.
- Iniciamos el recopilador de datos y esperamos unos segundos antes de detenerlo.
- Abrimos el informe correspondiente y observamos la información que recoge.

Una vez hecho esto, podemos consultar la información del sistema que se ha recogido durante la monitorización. Por ejemplo, en *Resultados del diagnóstico* podemos consultar información general del uso de los recursos de la máquina (ver 11), como porcentaje de uso de la CPU, número de entradas y salidas por segundo o tanto por ciento de la memoria utilizada hasta el momento.

Si lo que queremos es consultar información de la CPU, se han recopilado datos acerca del número medio, máximo y mínimo de cambios de contexto, envíos de excepciones, llamadas al sistema,... Por ejemplo, puede observarse que el número máximo de operaciones de escritura de archivos han sido 8 y

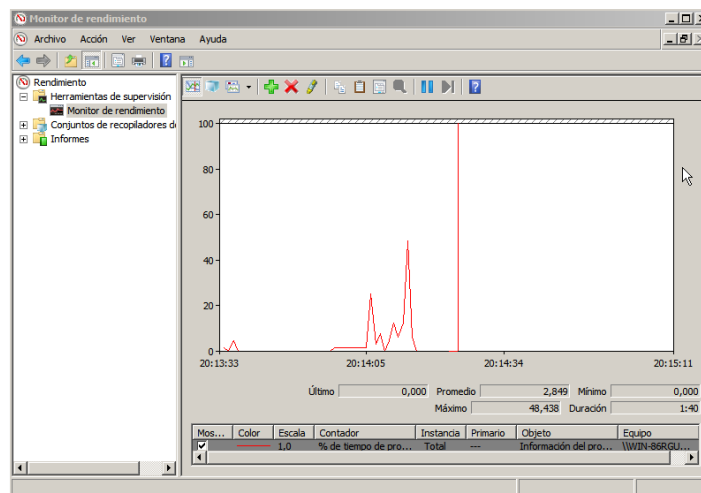


Figura 9: Herramientas de supervisión: monitor de rendimiento

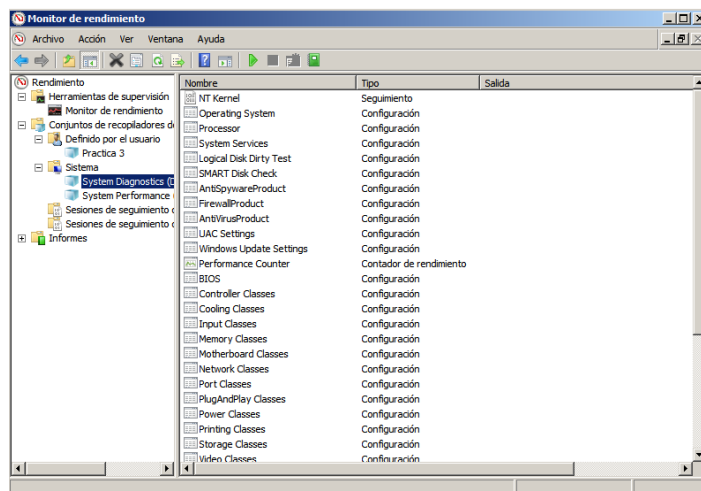


Figura 10: Recopiladores de datos del sistema

el número mínimo 0, mientras que la media ha sido de 2; o que el número de procesos se ha mantenido más o menos estable (ver 12).

Rendimiento			
Información general de recursos			
Componente	Estado	Uso	Detalles
CPU	Inactivo	9 %	Carga de CPU baja.
Red	Inactivo	0 %	El adaptador de red más ocupado es inferior al 15%.
Disco	Inactivo	12 /sec	La E/S de disco es inferior a 100 (lectura/escritura) por segundo en el disco 0.
Memoria	Normal	48 %	528 MB disponibles.

Figura 11: Información general de recursos

Sistema		Superior: 15 de 15		
counter		Medio	Mínimo	Máximo
% de cuota de Registro en uso		4	4	4
Cambios de contexto/s		283	119	600
Envíos de excepciones/s		0	0	0
Bytes de control de archivo/s		12,781	4,311	44,060
Operaciones de control de archivo/s		178	36	709
Operaciones con datos de archivo/s		22	1	43
Bytes de lectura de archivo/s		5,843	0	32,924
Operaciones de lectura de archivo/s		20	0	39
Bytes de escritura de archivo/s		43,564	0	132,064
Operaciones de escritura de archivo/s		2	0	8
Procesos		38	38	39
Longitud de la cola del procesador		0.667	0	4
Llamadas al sistema/s		4,290	1,957	12,411
Tiempo de actividad del sistema		3,197	3,193	3,201
Subprocesos		399	393	404

Figura 12: Información acerca de la CPU: sistema

5. Cree un recopilador de datos definido por el usuario (modo avanzado) que incluya tanto el contador de rendimiento como los datos de seguimiento.

Se puede seguir la creación del conjunto de recopiladores de datos en forma de capturas de pantalla de cada paso que se adjuntan en la sección 11.1, para que no interfieran con la lectura del resto del documento. A continuación se explican los pasos a seguir:

- Seleccionamos *Conjuntos de recopiladores de datos - Definido por el usuario*. Hacemos clic derecho y seleccionamos *Nuevo - Conjunto de recopiladores de datos*.
- Escribimos el nombre que queramos darle al conjunto y seleccionamos *Crear manualmente (avanzado)* (45). Le damos a *Siguiente*.
- En el tipo de datos a incluir, seleccionamos *Contador de rendimiento y Datos de seguimiento de eventos* (46). Pasamos al siguiente paso.
- En intervalo de muestra dejamos 15 segundos, y en contadores de rendimiento, añadimos todos los referentes al procesador, proceso y servicio web, tal y como indica en el enunciado (47).
- Le damos a *Siguiente* hasta que nos pregunte dónde guardar los datos. En el directorio raíz, ponemos el directorio **Escritorio/logs** y finalizamos el proceso seleccionando la opción de iniciar el recopilador de datos (48).

Una vez hecho esto, buscamos *Informes - Definido por el usuario* y seleccionamos el informe con el nombre del recopilador de datos que hemos creado. El resultado puede verse en la figura (13).

En este informe es difícil distinguir los datos de seguimiento, por lo que los analizamos por separado. En primer lugar, los referentes al servicio web (ver figura 14). Como no ha habido actividad del servidor web, todos los contadores están a cero.

Por otro lado, si seleccionamos los contadores referentes al proceso, podemos ver el siguiente resultado (15). En este caso ha habido mucha más actividad, aunque se pueden identificar algunos contadores. Por ejemplo, los picos azul marino corresponden al número de operaciones de E/S.

Otra forma que puede resultar más fácil de consultar es mostrando directamente los porcentajes en el informe (16). En esta ocasión se pueden consultar los datos referentes al procesador, como porcentaje de tiempo inactivo o el total de interrupciones, entre otros parámetros. Por ejemplo, puede verse que el procesador ha estado la mayor parte del tiempo de monitorización inactivo, con un 99,719 % o que se han producido un total de 69,400 interrupciones.

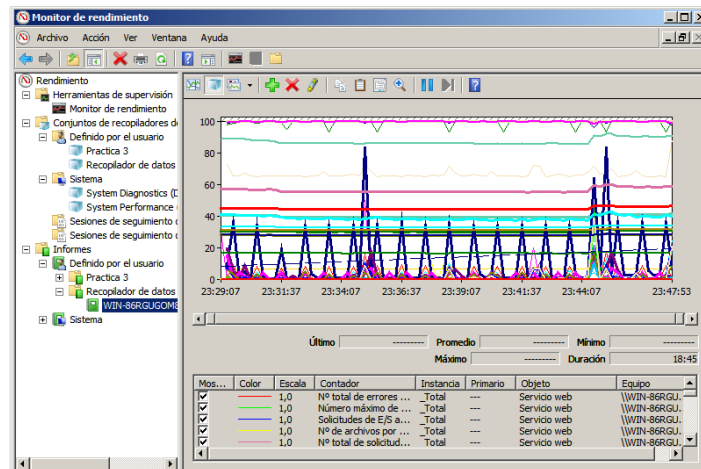


Figura 13: Informe de diagnóstico I

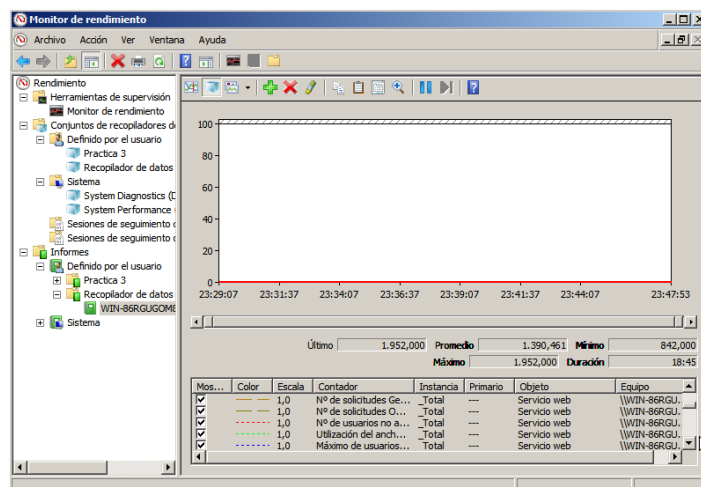


Figura 14: Datos de seguimiento referentes al servicio web

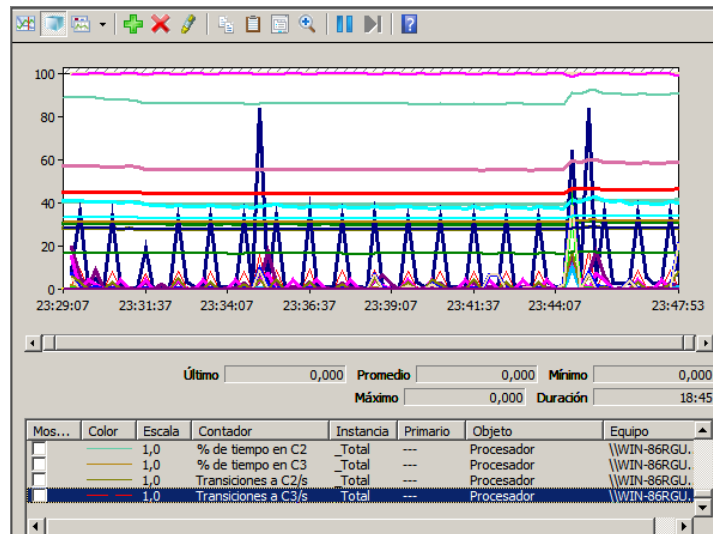


Figura 15: Datos de seguimiento referentes al proceso

\\WIN-86RGUGOM83L		
Procesador		_Total
% de tiempo de DPC		0,004
% de tiempo de interrupción		0,004
% de tiempo de procesador		0,281
% de tiempo de usuario		0,142
% de tiempo en C1		99,187
% de tiempo en C2		0,000
% de tiempo en C3		0,000
% de tiempo inactivo		99,719
% de tiempo privilegiado		0,139
DPC en cola/s		7,990
Interrupciones/s		69,400
Transiciones a C1/s		66,739
Transiciones a C2/s		0,000
Transiciones a C3/s		0,000
Velocidad de DPC		0,132
Proceso		_Total
% de tiempo de procesador		99,985
% de tiempo de usuario		0,135
% de tiempo privilegiado		99,850
Bytes de bloque no paginado		563.130,526
Bytes de bloque paginado		4.009.355,789
Bytes de datos ES/s		6.930,931
Bytes de escritura de ES/s		1.863,559
Bytes de lectura de ES/s		5.067,371
Bytes del archivo de paginación		279.157.059,368
Bytes privados		279.157.059,368

Figura 16: Informe de diagnóstico II

6. Instale alguno de los monitores comentados arriba en su máquina y pruebe a ejecutarlos (tenga en cuenta que si lo hace en la máquina virtual, los resultados pueden no ser realistas). Alternativamente, busque otros monitores para hardware comerciales o de código abierto para Windows y Linux.

`hddtemp` es una herramienta que permite consultar la temperatura de los discos duros. Instalamos la aplicación utilizando el gestor de paquetes [9].

- Como la instalación se está haciendo en una máquina con Ubuntu, utilizaremos el comando: `sudo apt-get install hddtemp`.
- Para que estén disponibles los sensores, necesitamos reiniciar el servicio `kmod`, que permite cargar los módulos: `sudo service kmod start`. Una vez hecho esto, ya estamos preparados para ejecutar el programa.
- La sintaxis es la siguiente: `sudo hddtemp <opciones><dispositivo>`. Hay una gran cantidad de opciones que pueden consultarse en el manual [10], por ejemplo, la opción `-u=C|F` se utiliza para que la salida por pantalla sea en grados Celsius o Fahrenheit, respectivamente.

Puede verse un ejemplo de uso en la captura (17).

```
minim@minim:~$ sudo hddtemp /dev/sda
/dev/sda: ST1000DL002-9TT153: 27°C
```

Figura 17: Consultando la temperatura del dispositivo `sda`

Se prueba ahora a instalar `xsensors`. No se detallará la instalación pues es tan simple como repetir los dos primeros pasos de la instalación anterior, utilizando el paquete `xsensors`. La diferencia entre estas dos utilidades es que esta vez puede accederse desde la barra de herramientas de Ubuntu, y que los resultados se muestran a través de una interfaz gráfica, como se puede ver en la captura (18).

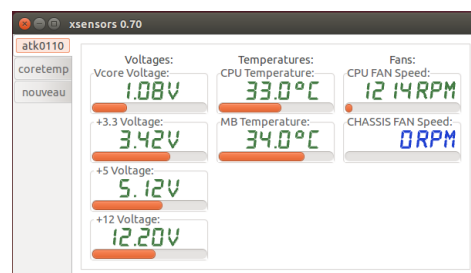


Figura 18: Interfaz gráfica de `xsensors`

Otros monitores

- **Open Hardware Monitor:** Se trata de un monitor de código abierto disponible para Windows y Linux en arquitecturas de 32 y 64 bits. Permite monitorizar temperatura, velocidad del ventilador y voltajes, entre otros parámetros [8].
- **HWMonitor:** Permite monitorizar temperatura, velocidad del ventilador y voltajes. Está disponible para Windows y se trata de una aplicación de pago [11].
- **psensor:** es una herramienta basada en `lm-sensors` y que nos permite monitorear la temperatura del hardware, a través de una interfaz gráfica (ver (19)), en la que pueden activarse los sensores que se deseen monitorizar [9].

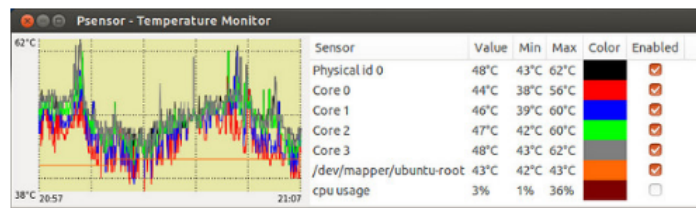


Figura 19: Interfaz gráfica de **psensor** [9]

7. 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.

Munin es una herramienta de monitorización que registra todo lo que ocurre dentro del servidor y genera estadísticas por día, semanas, meses y años en forma de gráficas a través de una interfaz web [12]. Podemos monitorizar por ejemplo, procesos de **apache** (20), consultas de **mysql** (21) o parámetros como disco, memoria, procesos (22),...

- Procesos de **apache**: en esta gráfica pueden consultarse el número de accesos por segundo que se han producido. Pueden comprobarse así cuáles son las horas de más acceso fijándonos en los picos de la gráfica. Por ejemplo, se puede ver cómo se produce un mayor número de accesos a las 11 de la noche y las 7 de la mañana.

Un detalle a notar es que en la parte inferior de las gráficas aparece la fecha de actualización de los datos que aparecen.

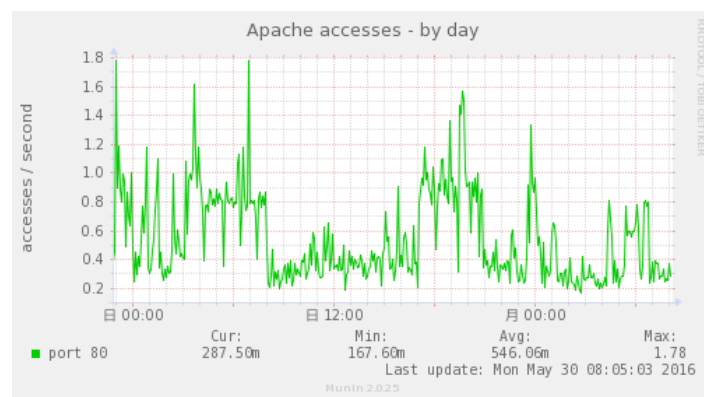


Figura 20: Monitorizando los recursos de un servidor: procesos de **apache**

- Consultas de **mysql**: en esta gráfica se observa una mayor complejidad que en la gráfica anterior. Esta vez pueden consultarse tanto el tipo de consulta que se ha hecho en **mysql** como los momentos en los que se produjeron un mayor número de consultas. Por ejemplo, puede verse que el momento en el que se produjo un mayor número de consultas (ver el color negro, **total**) fue cerca de las 00:00, y que por lo general, el tipo de operación que más se hace es **cache_hits**, seguido de las consultas de tipo **select**.
- Procesos: esta última gráfica muestra el número de procesos en distintos estados a lo largo del día. Puede verse cómo el número de procesos es más o menos uniforme. La mayoría de los

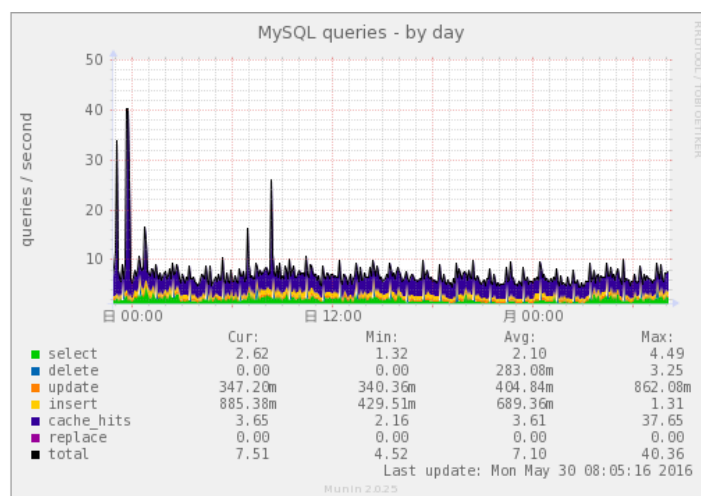


Figura 21: Monitorizando los recursos de un servidor: consultas de mysql

procesos se encuentran suspendidos (`sleeping`) y algunos de ellos preparados (`runnable`). El resto se mantiene en una tasa nula o prácticamente despreciable.

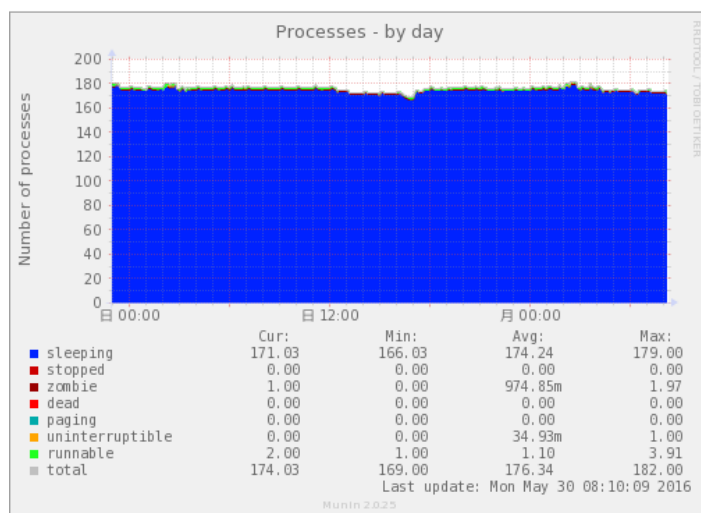


Figura 22: Monitorizando los recursos de un servidor: procesos

8. Escriba un breve resumen sobre alguno de los artículos donde se muestra el uso de `strace`, o busque otro programa similar y coméntelo.

Se ha consultado el artículo https://debian-administration.org/article/352/Using_strace_to_debug_application_errors.

`strace` es una herramienta para los sistemas operativos GNU/Linux, que sirve para monitorear las llamadas al sistema. Esto puede resultar muy útil, especialmente cuando se producen errores en un programa sin motivo aparente. A modo de ejemplo, en el artículo consultado se resuelve un error en un servidor gracias al uso de `strace` estudiando las llamadas al sistema de varios clientes.

Se puede instalar `strace` utilizando el gestor de paquetes: `sudo apt-get install strace`. La sintaxis

del comando es **strace** <opciones>program <argumentos>, donde <opciones> serán las opciones de **strace** y <argumentos> serán los argumentos que se le pasen al programa. Algunas opciones comunes de **strace** son **-o**, que permite redirigir la salida por pantalla a un fichero, o **-p**, que permite conectarse a un programa que esté ya ejecutándose.

Para iniciar el programa, ejecutamos **strace** <nombre-binario>, donde <nombre-binario> es el binario del fichero cuyas llamadas quieren monitorearse. La salida por pantalla contendrá, entre otras cosas, las llamadas al sistema junto al código **-1** si la llamada fracasa, por lo que será más sencillo encontrar el error en la ejecución del programa.

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).

Se accederá a la consola **mysql** desde la terminal, en la máquina virtual de Ubuntu Server [13].

- Comenzamos creando la base de datos. Para ello, ejecutamos **create database** <nombre>, donde <nombre> será el nombre de la base de datos a crear (23), y accedemos a ella [14].

```
mysql> create database prueba;
Query OK, 1 row affected (0.01 sec)

mysql> use prueba;
Database changed
```

Figura 23: Creación de la base de datos prueba

- Seguidamente, se introduce una tabla en dicha base de datos. Para ello, ejecutamos el comando **create table** <nombre>(<columnas>), donde <nombre> es el nombre que se le quiere dar a la tabla, y <columnas> incluye el nombre de las columnas, separadas por comas, junto con el tipo de dato que contiene [15]. Como se puede ver en (24), nuestra tabla tendrá una primera columna **n1** que contendrá enteros, una segunda columna **n2** que contendrá datos del mismo tipo, y por último, una columna **n3** con datos del tipo **varchar** de longitud 10.

```
mysql> create table tablaprueba(n1 int, n2 int , n3 varchar(10) );
Query OK, 0 rows affected (0.05 sec)
```

Figura 24: Creación de la tabla tablaprueba

- Después de crear la tabla, añadimos datos sobre los que realizar la consulta. La sintaxis es **insert into** <nombre-tabla>values <values>, donde <values> son los valores que se le quiere dar a la tabla, entre paréntesis y ordenados por columnas [16]. Ver (25).

```
mysql> insert into tablaprueba values (1,2,'tres');
Query OK, 1 row affected (0.02 sec)

mysql> insert into tablaprueba values (2,4,'seis');
Query OK, 1 row affected (0.01 sec)

mysql> insert into tablaprueba values (4,4,'ocho');
Query OK, 1 row affected (0.02 sec)
```

Figura 25: Añadiendo datos a la tabla tablaprueba

- Hacemos algunas consultas adicionales y ejecutamos **show profile**. Con esto puede verse una lista de las consultas que se le han enviado al servidor. El resultado puede verse en (26), donde pueden verse también algunas consultas fallidas.

```
mysql> show profiles;
+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+
| 1 | 0.00504550 | create dataabse prueba |
| 2 | 0.00591600 | create database prueba |
| 3 | 0.01139800 | SELECT DATABASE() |
| 4 | 0.04087150 | show databases |
| 5 | 0.00376200 | show tables |
| 6 | 0.00013950 | create table tablaprueba(int n1, int n2, varchar(10)) |
| 7 | 0.04592675 | create table tablaprueba(n1 int, n2 int , n3 varchar(10) ) |
| 8 | 0.00010500 | insert into tablaprueba(1,2,'tres') |
| 9 | 0.01925550 | insert into tablaprueba values (1,2,'tres') |
| 10 | 0.00111975 | insert into tablaprueba (2,4,'seis') |
| 11 | 0.01400100 | insert into tablaprueba values (2,4,'seis') |
| 12 | 0.01693025 | insert into tablaprueba values (4,4,'ocho') |
| 13 | 0.01225925 | show tables |
| 14 | 0.01503550 | select n3 from tablaprueba where n2=4 |
+-----+-----+-----+
14 rows in set (0.00 sec)
```

Figura 26: Ejecución de `show profile`

- Se muestra el profile de una de las consultas, en concreto de la consulta 14 (27). El resultado es una lista de los distintos procesos que se han ejecutado durante la consulta y sus tiempos. Por ejemplo, en este caso, podemos ver como el proceso que ha consumido más tiempo es `checking query cache for query`, es decir, comprobar si la respuesta a la consulta se encuentra en la caché [17]. Hay que tener en cuenta que para ver cuál es el proceso en el que se ha consumido más tiempo es necesario sumar los tiempos de todas las veces que se ejecute (por ejemplo, `waiting for query cache lock`, entre otras, aparece varias veces en el profile).

```
mysql> show profile for query 14;
+-----+-----+-----+
| Status | Duration |
+-----+-----+-----+
| starting | 0.000023 |
| Waiting for query cache lock | 0.000004 |
| checking query cache for query | 0.009556 |
| checking permissions | 0.000020 |
| Opening tables | 0.000025 |
| System lock | 0.000011 |
| Waiting for query cache lock | 0.002167 |
| init | 0.001386 |
| optimizing | 0.001557 |
| statistics | 0.000022 |
| preparing | 0.000017 |
| executing | 0.000003 |
| Sending data | 0.000185 |
| end | 0.000006 |
| query end | 0.000005 |
| closing tables | 0.000008 |
| freeing items | 0.000007 |
| Waiting for query cache lock | 0.000002 |
| freeing items | 0.000024 |
| Waiting for query cache lock | 0.000002 |
| freeing items | 0.000001 |
| storing result in query cache | 0.000003 |
| logging slow query | 0.000002 |
| cleaning up | 0.000002 |
+-----+-----+-----+
24 rows in set (0.00 sec)
```

Figura 27: Profile de la consulta 14

10. Cuestiones opcionales

10.1. Cuestión opcional 2. Instale Nagios en su sistema documentando el proceso y muestre el resultado de la monitorización de su sistema comentando qué aparece.

Instalación de Nagios en CentOS: el proceso se ha hecho siguiendo un tutorial que puede encontrarse en la página oficial de Nagios [18]. Es importante notar que todo el proceso se ha seguido como root para evitar problemas con los permisos.

En primer lugar, se instalan todos aquellos programas que harán falta para la instalación:

```
yum install httpd php php-cli gcc glibc glibc-common gd gd-devel net-snmp openssl-devel  
wget unzip -y
```

Una vez hecho esto, se crea un usuario y grupo para Nagios:

```
useradd nagios  
groupadd nagcmd
```

Modificamos la cuenta del usuario haciendo uso de `usermod` [19]:

Una vez hecho esto, se crea un usuario y grupo para Nagios, y hacemos que los usuarios `nagios` y `apache` pertenezcan a dicho grupo:

```
usermod -a -G nagcmd nagios  
usermod -a -G nagcmd apache
```

Una vez llegados a este punto, se descargan y extraen los `.tar.gz` necesarios para Nagios y sus plugins, y se compilan los archivos ejecutando las siguientes líneas de comandos: `cd /tmp`

```
wget https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.1.1.tar.gz
```

```
wget http://www.nagios-plugins.org/download/nagios-plugins-2.1.1.tar.gz
```

```
tar xzf nagios-4.1.1.tar.gz
```

```
tar xzf nagios-plugins-2.1.1.tar.gz
```

```
cd nagios-4.1.1
```

```
./configure --with-command-group=nagcmd
```

```
make all
```

```
make install
```

```
make install-init
```

```
make install-config
```

```
make install-commandmode
```

```
make install-webconf
```

A continuación se cambia la contraseña por defecto del usuario `nagiosadmin`, que nos servirá para acceder a la aplicación a través de la interfaz web.

```
htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
```

Una vez hecho esto, nos pedirá la contraseña que se quiere introducir. Por último, se instalan los plugins:

```
cd /tmp/nagios-plugins-2.1.1
```

```
./configure --with-nagios-user=nagios --with-nagios-group=nagios --with-openssl
```

```
make all
```

```
make install
```

Para terminar, iniciamos el servicio **httpd** y **nagios** y podremos acceder a la herramienta escribiendo en el navegador web localhost/nagios, tal y como se puede ver en la captura (28).

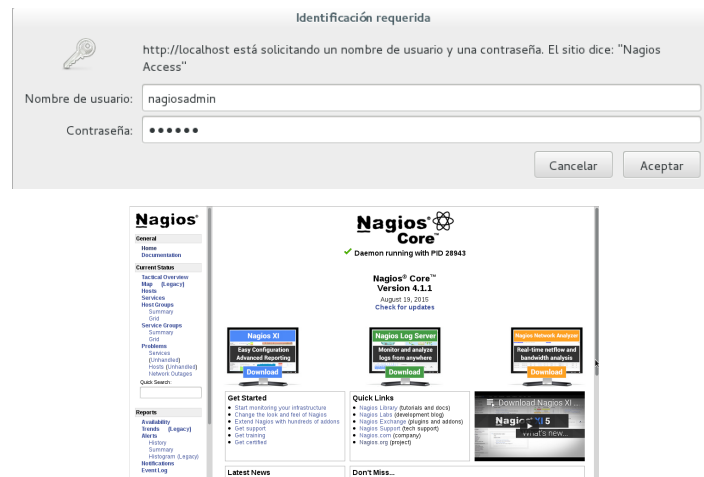


Figura 28: Accediendo a Nagios desde CentOS

Instalación en Ubuntu Server. La instalación en Ubuntu Server utilizando el gestor de paquetes es mucho más rápida y simple que la instalación anterior [21]. Se seguirán los siguientes pasos:

- Ejecutamos la línea de comandos `sudo apt-get install nagios3`
- Nos saldrá una pantalla como la que puede verse en la figura (29). Seleccionamos *Sitio de Internet*.
- En el siguiente paso se pedirá un nombre para el sistema de correo (30).
- Se pedirá una contraseña para el usuario anterior, y una vez hecho esto, la instalación finalizará.
- Una vez hecho esto, podremos acceder del mismo modo que se hizo en CentOS, accediendo desde el navegador web a localhost/nagios3/ (ver (31)), con el usuario **nagiosadmin** y la contraseña que se estableció en el paso anterior.

Probando la monitorización del sistema: Nagios permite monitorizar algunos servicios tales como HTTP, SSH, IMAP,... entre otros. Si el servicio es privado (por ejemplo, uso de CPU, memoria de un host), no se puede monitorizar con Nagios sin usar un agente intermedio [20].

Una vez se ha accedido a Nagios, hacemos clic en la opción **Hosts**, que muestra los detalles de todos los hosts del servidor. En este caso sólo podemos ver los detalles de **localhost** (32). Puede verse cómo el host está activo (UP) y activo desde hace 18 minutos, aproximadamente.

Si accedemos a la opción **Services**, en la imagen (33) podemos ver por una parte la misma información que en el apartado anterior (hay tan sólo un host, y éste está activo), y por otra parte, el estado

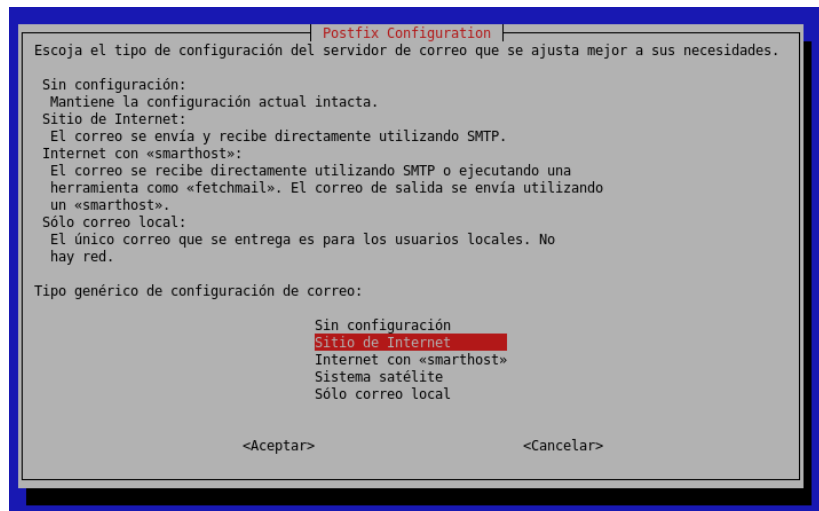


Figura 29: Instalando Nagios en Ubuntu Server I

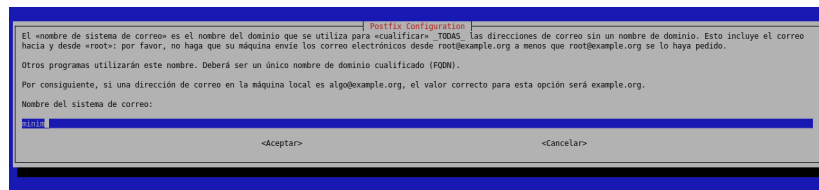


Figura 30: Instalando Nagios en Ubuntu Server II

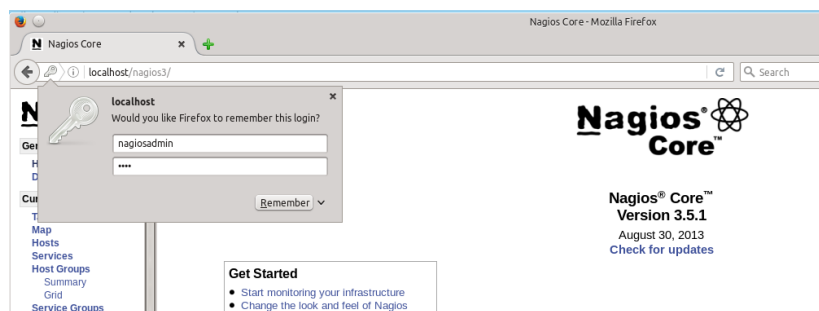


Figura 31: Accediendo a Nagios desde Ubuntu Server

Host Status Details For All Host Groups				
Host ♦♦	Status ♦♦	Last Check ♦♦	Duration ♦♦	Status Information
localhost	UP	2016-05-30 12:52:49	0d 0h 18m 45s	ECO OK - Paquetes perdidos = 0%, RTA = 0.03 ms

Figura 32: Monitorizando con Ganglia I

de los servicios que se monitorizan. Puede observarse que hay cinco servicios funcionando correctamente y una advertencia, que se corresponde con el problema en **All Problems**. En **All Types** puede consultarse el número total de servicios monitorizados que, como puede deducirse es 6.

Si queremos obtener más información acerca de la advertencia, accedemos al host desde esa misma

Host Status Totals					Service Status Totals				
Up	Down	Unreachable	Pending		Ok	Warning	Unknown	Critical	Pending
1	0	0	0		5	1	0	0	0
All Problems All Types					All Problems All Types				
0					1 6				

Figura 33: Monitorizando con Ganglia II

página, haciendo clic en su nombre. Se muestra un listado de los servicios con su estado, tiempo de duración, última actualización,... entre otros datos (34). El servicio que nos interesa es el que está en color amarillo, el espacio en disco, que se corresponde con la advertencia.

Host ♦♦	Service ♦♦	Status ♦♦	Last Check ♦♦	Duration ♦♦	Attempt ♦♦	Status Information
localhost	Current Load	OK	2016-05-30 12:48:09	0d 0h 12m 29s	1/4	OK - carga media: 0.07, 0.15, 0.22
	Current Users	OK	2016-05-30 12:48:59	0d 0h 11m 39s	1/4	USUARIOS OK - 3 usuarios actualmente en
	Disk Space	WARNING	2016-05-30 12:47:49	0d 0h 10m 49s	4/4	DISK WARNING - free space: / 613 MB (11% inode=49%)
	HTTP	OK	2016-05-30 12:45:39	0d 0h 9m 59s	1/4	HTTP OK: HTTP/1.1 200 OK - 937 bytes en 0.001 segundo tiempo de respuesta
	SSH	OK	2016-05-30 12:46:29	0d 0h 9m 9s	1/4	SSH OK - OpenSSH_6.6.1p1 Ubuntu-Zubuntu2.7 (protocol 2.0)
	Total Processes	OK	2016-05-30 12:47:19	0d 0h 8m 19s	1/4	PROCS ACCEPTAR: 186 procesos

Figura 34: Monitorizando con Ganglia III

Por último, observar que si se quiere obtener más información acerca de un proceso, tan sólo tenemos que hacer clic en su nombre. Haciendo esto, obtendremos un panel de información como el que aparece en la figura (35).

Service State Information	
Current Status:	OK (for 0d 0h 12m 0s)
Status Information:	USUARIOS OK - 3 usuarios actualmente en
Performance Data:	users=3;20;50;0
Current Attempt:	1/4 (HARD state)
Last Check Time:	2016-05-30 12:48:59
Check Type:	ACTIVE
Check Latency / Duration:	0.207 / 0.005 seconds
Next Scheduled Check:	2016-05-30 12:53:59
Last State Change:	2016-05-30 12:38:59
Last Notification:	N/A (notification 0)
Is This Service Flapping?	NO (0.00% state change)
In Scheduled Downtime?	NO
Last Update:	2016-05-30 12:50:59 (0d 0h 0m 0s ago)
Active Checks:	ENABLED
Passive Checks:	ENABLED
Obsessing:	ENABLED
Notifications:	ENABLED
Event Handler:	ENABLED
Flap Detection:	ENABLED

Figura 35: Monitorizando con Ganglia IV

10.2. Cuestión opcional 3. Haga lo mismo que con Munin.

- En primer lugar, analizamos un pequeño resumen que nos muestra información del servidor que estamos monitoreando, en este caso, Wikimedia (36). El campo **CPUs Total** se corresponde con el número de CPUs virtuales asignadas al servidor [22]. Puede observarse que en la última hora (la fecha de actualización aparece en el campo **localtime**), la utilización ha sido del 12 % de media. Puede verse también que el número de hosts funcionando es de 1117, mientras que hay 31 host caídos o inaccesibles.

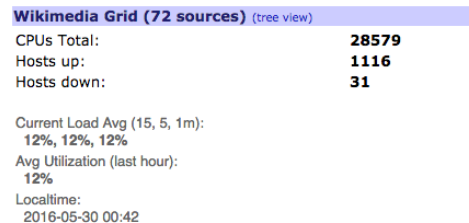


Figura 36: Monitorizando con Ganglia

- Una de las gráficas que pueden consultarse es la del uso de memoria (37). Puede verse como en el momento en el que se tomaron los datos, la memoria total empleada fue de 86.3 T, siendo la mayor parte de ella consumida en la memoria usada y la memoria caché. Usando el mínimo y el máximo pueden consultarse los picos de uso de memoria, aunque como puede verse, durante la última hora ha sido bastante homogéneo.

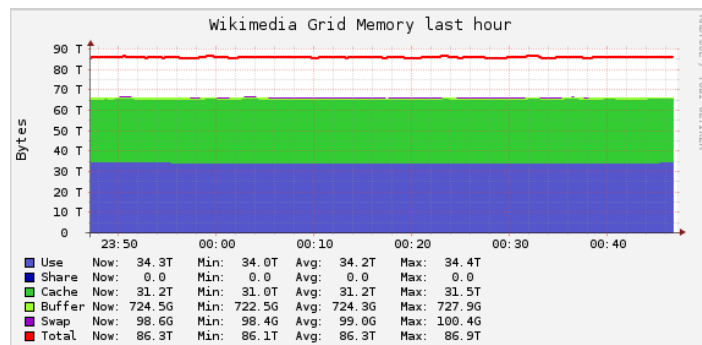


Figura 37: Monitorizando con Ganglia: uso de memoria

- También puede consultarse el uso de la CPU durante la última hora (38). Puede verse que ha estado ociosa la mayor parte del tiempo (de media durante la última hora, el 90.9 % del tiempo), o que el tiempo consumido por el usuario ha sido del 6.9 %.

El valor *nice* es la prioridad de un proceso, siendo ésta mayor cuanto menor sea el valor *nice*, que se mueve en el rango [-20,19]. En la gráfica puede observarse cómo el 0.3 % de uso de la CPU en la última hora ha sido para la ejecución de procesos con valor *nice* positivo, esto es, de prioridad moderada o baja [23].

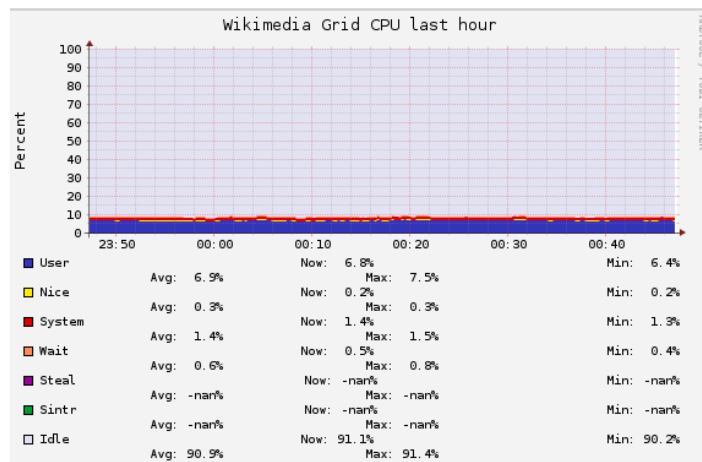


Figura 38: Monitorizando con Ganglia: uso de la CPU

10.3. Cuestión opcional 5: Pruebe a instalar este monitor en alguno de sus tres sistemas. Realice capturas de pantalla del proceso de instalación y comente capturas de pantalla del proceso en ejecución.

Cacti puede instalarse a través del gestor de paquetes [24]. La instalación se detalla a continuación acompañada de capturas de pantalla:

- Ejecutamos `sudo apt-get install cacti`.
- Nos saldrá una pantalla como la que aparece en la captura (39). En este caso seleccionamos *Sí* a la configuración de la base de datos con `dbconfig-common`.

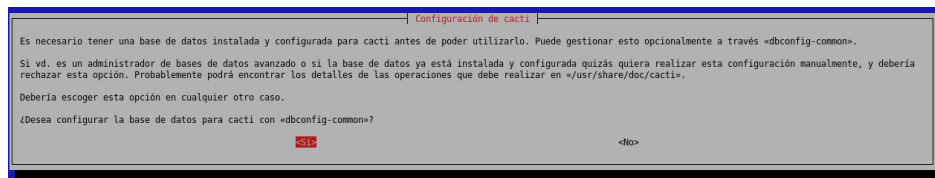


Figura 39: Instalación de Cacti I

- Si no se ha hecho antes, establecemos la contraseña para el usuario `root` en MySQL
- En el siguiente paso, se selecciona el servidor web. En este caso utilizaremos `apache2` (40). Con este paso acaba la instalación.

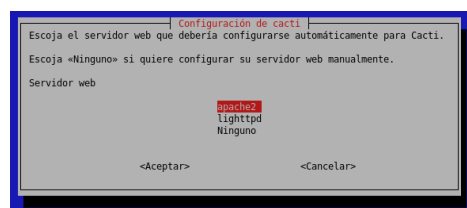
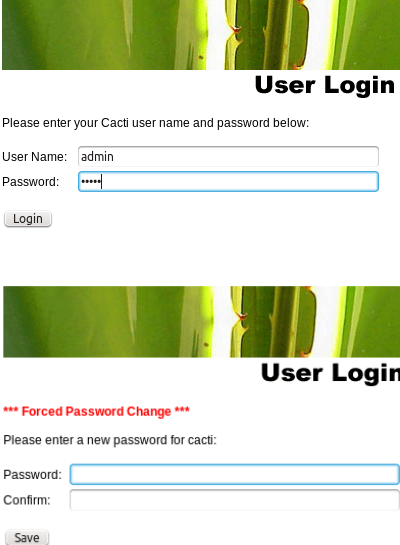


Figura 40: Instalación de Cacti II

- Tratamos ahora de acceder desde el navegador web de la misma manera que en ejercicios anteriores: `localhost/cacti`. Si es la primera vez que se accede, Cacti pedirá algunas rutas de archivos de configuración antes de pasar a la autenticación. Para ello, tendremos que introducir el usuario y contraseña por defecto `admin`, `admin`. Cacti forzará el cambio de contraseña inmediatamente por motivos de seguridad (41).



User Login

Please enter your Cacti user name and password below:

User Name:

Password:

User Login

*** Forced Password Change ***

Please enter a new password for cacti:

Password:

Confirm:

Figura 41: Autenticación en Cacti

- Una vez seguidos estos pasos, debería aparecernos el panel principal de Cacti como vemos en la captura (42).

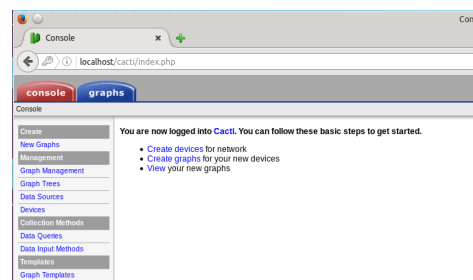


Figura 42: Accediendo a Cacti

Hecho esto, si todo ha salido bien, haciendo clic en la pestaña *Gráficas* deberíamos poder observar actividad desde el primer momento. Si esto no ocurre, puede ser interesante consultar los tiempos de actualización en la configuración.

A partir de aquí, las opciones son similares a las disponibles en los ejercicios anteriores. Hay una serie de gráficas creadas por defecto, aunque pueden crearse nuevas que monitoricen distintos parámetros en la opción *New Graphs*. Pueden analizarse las gráficas de la misma forma que en ejercicios anteriores, por lo que no se profundizará demasiado en este aspecto. Por ejemplo, en la imagen (44) puede verse que en el momento en el que se actualizaron los datos, había 197 procesos ejecutándose, aunque el máximo de procesos ejecutándose en un mismo momento fueron 200, y la media de procesos ejecutados por unidad de tiempo es de 198.

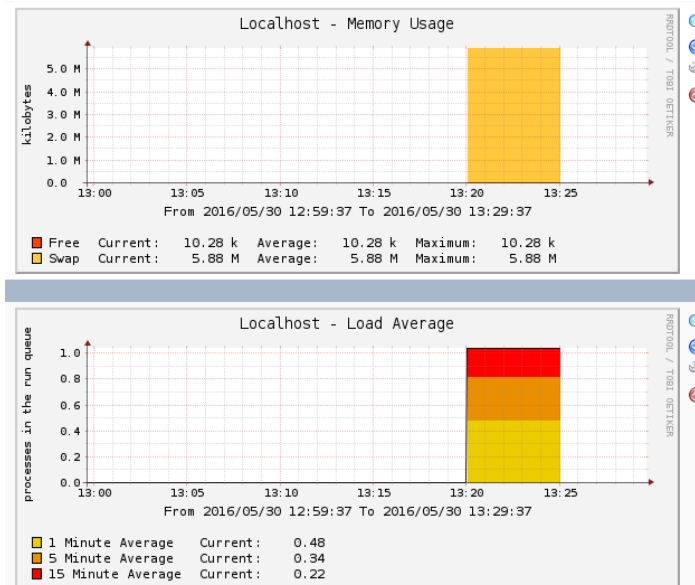


Figura 43: Monitorizando con Cacti: uso de la memoria y carga

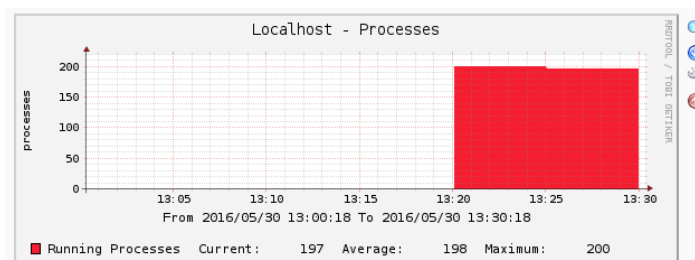


Figura 44: Monitorización con Cacti: procesos

11. Capturas de pantalla

11.1. Cuestión 5

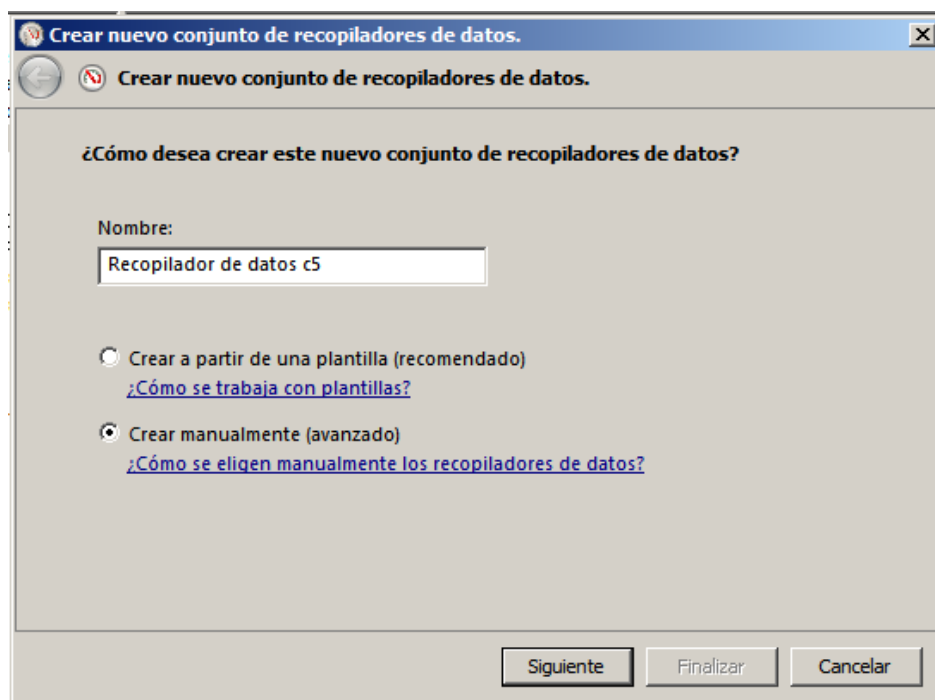


Figura 45: Creación de un recopilador de datos definido por el usuario (modo avanzado)

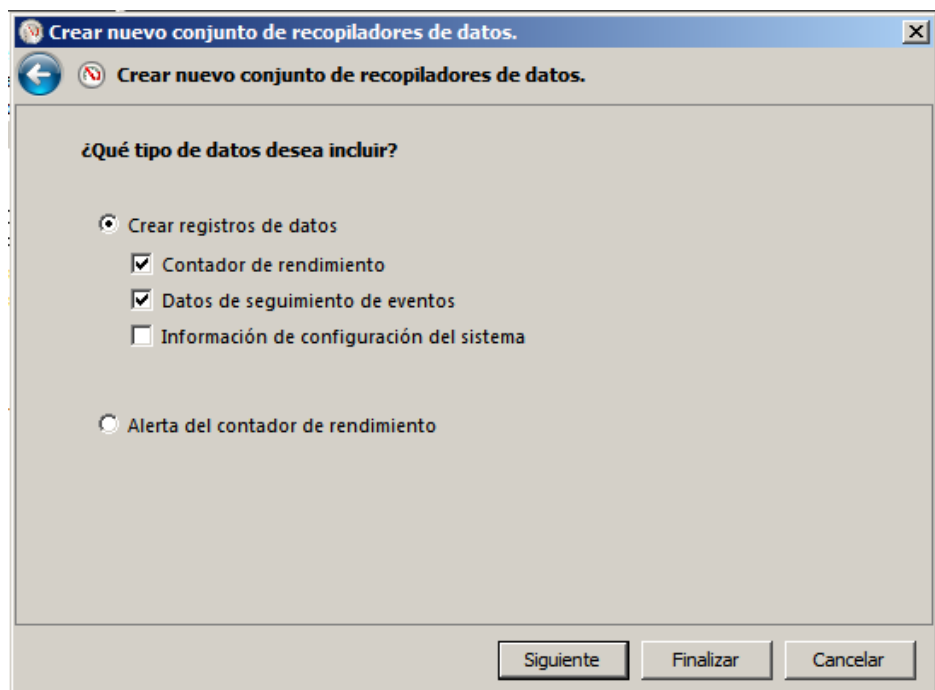


Figura 46: Incluir el contador de rendimiento y los datos de seguimiento

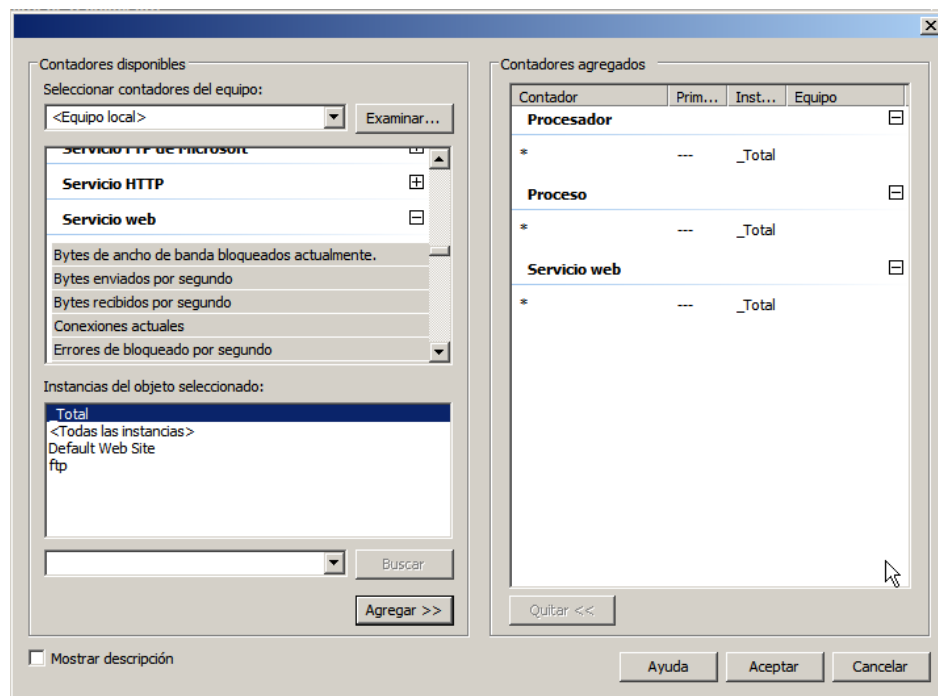


Figura 47: Datos de seguimiento: procesador, proceso, servicio web

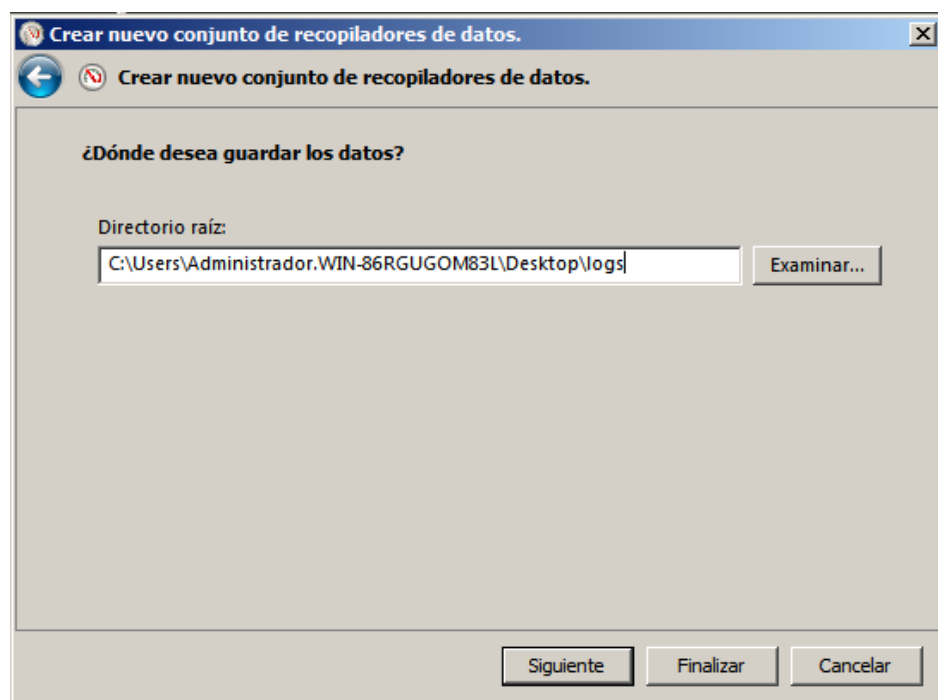


Figura 48: Almacenar el resultado en el directorio Escritorio/logs

Referencias

- [1] <http://askubuntu.com/questions/21657/how-do-i-show-apt-get-package-management-history-via-command>
- [2] <http://unix.stackexchange.com/questions/105413/debian-what-is-var-log-apt-term-log-good-for>
- [3] <http://linux.die.net/man/1/zcat>
- [4] <http://linux.die.net/man/5/crontab>
- [5] <https://geekytheory.com/programar-tareas-en-linux-usando-crontab/>
- [6] <http://man7.org/linux/man-pages/man1/dmesg.1.html>
- [7] <http://www.linuxnix.com/what-is-linuxunix-dmesg-command-and-how-to-use-it/>
- [8] <http://openhwaremonitor.org/>
- [9] <http://xmodulo.com/monitor-system-temperature-linux.html>
- [10] <http://linux.die.net/man/8/hddtemp>
- [11] <http://www.cpubid.com/software/hwmonitor.html>
- [12] <http://munin-monitoring.org/>
- [13] <http://dev.mysql.com/doc/refman/5.7/en/show-profile.html>
- [14] <http://dev.mysql.com/doc/refman/5.7/en/database-use.html>
- [15] <http://dev.mysql.com/doc/refman/5.7/en/creating-tables.html>
- [16] <http://dev.mysql.com/doc/refman/5.7/en/loading-tables.html>
- [17] <https://dev.mysql.com/doc/refman/5.7/en/query-cache-thread-states.html>
- [18] <https://assets.nagios.com/downloads/nagioscore/docs/Nagios-Core-Installing-On-Centos7.pdf>
- [19] <http://www.debianadmin.com/users-and-groups-administration-in-linux.html#more-106>
- [20] <https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/monitoring-publicservices.html>
- [21] <https://help.ubuntu.com/community/Nagios3>
- [22] <https://sourceforge.net/p/ganglia/mailman/message/30539671/>
- [23] <http://serverfault.com/questions/116950/what-does-nice-mean-on-cpu-utilization-graphs>
- [24] <http://www.digitalocean.com/community/tutorials/installing-the-cacti-server-monitor-on-ubuntu>