

Monitorización de servicios

Cuestiones de la Práctica 3

LOTHAR SOTO PALMA
Universidad de Granada
1 de mayo de 2015

Índice

Información	1
Cuestión 1	1
Cuestión 2	2
Cuestión 3	4
Cuestión 4	5
Cuestión 5	6
Cuestión 6	11
Cuestión 7	18

Índice de figuras

1.	Registro con todo los discos correctos.	2
2.	Producción de fallo al disco /dev/sdb1.	2
3.	Registro indicando el fallo de sdb1.	3
4.	Quitando el disco del sistema.	3
5.	Registro sin el disco sdb1.	3
6.	Reposición del disco.	3
7.	Registro mostrando la recuperación del disco.	4
8.	Estado final de los discos de sistema.	4
9.	Pantalla de inicio Perfmon.	7
10.	Monitor de rendimiento Perfmon.	7
11.	Creación nuevo conjunto de recopiladores.	8
12.	Añadiendo registros de contador de rendimiento y datos de seguimiento de eventos.	8
13.	Añadiendo contadores: Procesador, proceso, servicio web.	9
14.	Estableciendo intervalo de muestra.	9
15.	Estableciendo carpeta log.	10
16.	Carpeta log.	10
17.	Monitor de rendimiento con los datos cada 15 segundos.	11
18.	Nagios instalado funcionando en CentOS.	13
19.	Estado de todos los hosts.	13
20.	Estados de los servicios del host.	14
21.	Estado del servicio HTTP.	14
22.	Nagios configurado para que muestre información de pnp4nagios.	16
23.	Estado del host, paquetes perdidos.	16
24.	Carga del sistema del Host.	17
25.	Estado del servicio PING.	17
26.	Información extensa sobre la carga del sistema.	18
27.	Activando el profiler.	19
28.	Creación de la BD Practicas.	19
29.	Selección base de datos Practicas.	19

30. Creacion de las tablas.	19
31. Estado profiler despues de crear tablas I.	20
32. Estado profiler despues de crear tablas II.	20
33. Estado profiler despues de insertar en las tablas.	21
34. Tiempos de creación de una tabla.	21
35. Tiempos de alteración de una tabla.	22
36. Tiempo total de creación de una tabla.	22
37. Consulta I.	23
38. Tiempos de la consulta I.	23
39. Consulta II.	23
40. Tiempos de la consulta II.	24
41. Tiempo total de las consultas.	24

Información

La práctica se ha realizado en maquinas virtuales centOS, Ubuntu Server y Windows Server 2012. Hay un error visual en los directorios en la cuestión 3, aparecen encima de los simbolos /. Todos los códigos son propios, además en la cuestión 7 la base de datos se ha implementado como se hizo en la asignatura Fundamentos de Bases de datos de la Universidad de Granada.

Cuestión 1

- ¿En qué archivos se guarda registro de los paquetes instalados en sistema con los gestores de paquetes de Ubuntu y centOS? Durante la práctica 2 instaló LAMP como un único paquete o instalando cada componente diferenciado. Busque en el archivo de registro las líneas correspondientes a la instalación y preséntelas.
- En el directorio `/var/log` es común encontrar archivos con extensiones en formato `<numero>.gz`. Por ejemplo, `.1.gz`. Explique como se generan estos archivos y que relación guardan entre ellos.

Solución:

- En el caso de centOS el archivo que guarda los paquetes que han sido instalados en el sistema se encuentra en `/var/log` y es llamado `yum.log`, que contiene los paquetes que han sido instalados y eliminados con la herramienta yum. En el caso de Ubuntu este registro se encuentra en el mismo directorio con el nombre `dpkg.log` y también muestra los paquetes instalados y borrados del sistema. En mi caso instalé cada uno de los componentes de LAMP por separado, de todas formas nos fijamos en el archivo `yum.log` que contiene los componentes de LAMP:

```

Mar 28 18:12:51 Installed: httpd-2.4.6-19.el7.centos.x86_64
Mar 28 18:18:43 Installed: mysql-community-libs-5.6.23-3.el7.x86_64
Mar 28 18:23:28 Installed: php-pdo-5.4.16-23.el7_0.3.x86_64
Mar 28 18:23:29 Installed: php-cli-5.4.16-23.el7_0.3.x86_64
Mar 28 18:23:29 Installed: php-5.4.16-23.el7_0.3.x86_64
Mar 28 18:23:29 Installed: php-mysql-5.4.16-23.el7_0.3.x86_64
...

```

Como podemos observar tenemos todos los paquetes que conforman LAMP.

- b) [1] Esos archivos se generan por un proceso de rotación del sistema, normalmente es *logrotate*, es decir, este programa es una tarea programada con la herramienta cron que se ejecuta diariamente, se encargará de dar una rotación de manera automática que comprime, y borra los registros de manera diaria, semanal, mensual ... Dependiendo de la configuración que se posea, ya que esta se puede modificar. Por lo que esos archivos son versiones más antiguas del registro, comprimidas y el número del archivo con el formato <numero>.gz es el número de generación de archivo.

Cuestión 2

Indique los pasos que ha seguido, comandos empleados y significado de los mismos. Junto a cada comando, presente las líneas del registro del RAID que son significativas en cada paso: indicación de fallo, reemplazo, inicio y finalización de la reconstrucción del RAID.

Solución:

[7] En primer lugar se ha utilizado la máquina virtual de Ubuntu server de la práctica 1 y se ha usado como ayuda la referencia [8] para finalizar la cuestión. Debido a que no tiene interfaz gráfica me ha sido más cómodo usar directamente el comando *cat* que monitorizarlo pero el resultado sería el mostrado. Primero revisamos el estado del registro (figura 1):

```

root@ubuntuserver:/home/lot94# cat /proc/mdstat
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdb1[1] sda1[0]
      8382400 blocks super 1.2 [2/2] [UU]

unused devices: <none>

```

Figura 1: Registro con todo los discos correctos.

Ahora vamos a producir un error en el disco con el comando "*mdadm --manage /dev/md0 --fail /dev/sdb1*", este comando nos permite administrar dispositivos que conformen un RAID, en específico el comando con la opción *--manage* nos permite añadir, quitar o incluso producir un fallo en el disco con la opción *--fail* (figura 2):

```

root@ubuntuserver:/home/lot94# mdadm --manage /dev/md0 --fail /dev/sdb1
[ 804.394979] md/raid1:md0: Disk failure on sdb1, disabling device.
[ 804.394979] md/raid1:md0: Operation continuing on 1 devices.
mdadm: set /dev/sdb1 faulty in /dev/md0

```

Figura 2: Producción de fallo al disco /dev/sdb1.

Volvemos a comprobar el registro y damos cuenta de que el disco sdb1 está marcado con una (F) que indica se ha producido un fallo en él. Otra diferencia es [2/1][U_] anteriormente se marcaba como [2/2][UU] esto es muestra de que uno de los discos esta "degradado" (figura 3):

```
root@ubuntuuserver:/home/lot94# cat /proc/mdstat
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdb1[1](F) sda1[0]
      8382400 blocks super 1.2 [2/1] [U_]

unused devices: <none>
```

Figura 3: Registro indicando el fallo de sdb1.

Para quitar el disco que ha fallado del sistema hacemos uso del comando *"mdadm --manage /dev/md0 --remove /dev/sdb1"* que nos permite exactamente eso, deshabilitar el disco (figura 4) y volvemos a comprobar el registro (figura 5):

```
root@ubuntuuserver:/home/lot94# mdadm --manage /dev/md0 --remove /dev/sdb1
mdadm: hot removed /dev/sdb1 from /dev/md0
```

Figura 4: Quitando el disco del sistema.

```
root@ubuntuuserver:/home/lot94# cat /proc/mdstat
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sda1[0]
      8382400 blocks super 1.2 [2/1] [U_]

unused devices: <none>
```

Figura 5: Registro sin el disco sdb1.

En este momento es cuando se tiene que reponer el disco por otro que este vacío, en un sistema real se tendría que sustituir el disco por otro y automáticamente se iniciaría el proceso de recuperación en nuestro caso añadimos un disco al sistema del mismo tamaño con el comando *"mdadm --manage /dev/md0 --add /dev/sdb1"* y verificamos que se produce la recuperación (figura 6-7):

```
root@ubuntuuserver:/home/lot94# mdadm --manage /dev/md0 --add /dev/sdb1
mdadm: added /dev/sdb1
```

Figura 6: Reposición del disco.

```

root@ubuntuserver:/home/lot94# cat /proc/mdstat
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdb1[2] sda1[0]
      8382400 blocks super 1.2 [2/1] [U_]
      [>.....] recovery = 0.3% (29312/8382400) finish=9.4min speed=14656K/sec
unused devices: <none>

```

Figura 7: Registro mostrando la recuperación del disco.

Por último se muestra el estado final del *RAID* ya con el disco sustituido y recuperado como podemos apreciar ahora *[2/2][UU]*, es decir no hay error en los discos están los dos operativos y parece que *sdb1[2]* se indica con un 2 por que se ha producido una sustitución de disco (figura 8):

```

root@ubuntuserver:/home/lot94# cat /proc/mdstat
Personalities : [raid1] [linear] [multipath] [raid0] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdb1[2] sda1[0]
      8382400 blocks super 1.2 [2/2] [UU]
unused devices: <none>

```

Figura 8: Estado final de los discos de sistema.

Cuestión 3

Añada a la configuración de cron una tarea que se ejecute diariamente y que copie una vez al día el contenido del directorio *~/codigo* a *~/seguridad/\$fecha* donde *\$fecha* es la fecha actual del sistema (puede usar el comando *date*). Otra tarea, se ejecutará una vez al mes y reunirá todos los directorios diarios creados para el mes pasado en un archivo *~/seguridad/dirCodigo.<numero>.gz*. Presente las líneas de configuración de cron afectadas, explicando su significado. Si crea ficheros por lotes, presente y explique el código.

Solución:

En primer lugar vamos a crear las copias diarias para ello creamos un script llamado *copiar_diaria.sh* con el siguiente contenido:

```

#!/bin/bash
fecha='date +%m-%d-%y'
directorio_origen=~/codigo/*
directorio_final=~/seguridad/$fecha
mkdir $directorio_final
cp $directorio_origen $directorio_final

```

El script obtiene la fecha del día con el comando *date* y copia el contenido del directorio *~/codigo* y lo pega en el directorio *~/seguridad/\$fecha*, es decir crea un directorio con la fecha actual y pega los archivos en dicha carpeta. Seguido de esto creamos otro script llamado *compresion_mensual.sh* que se va a encargar de realizar las compresiones:

```

#!/bin/bash
directorio=./
cd ~/seguridad

```

```

numero='ls -l dirCodigo.?.tgz | wc -l'
numero=$((numero+1))
nombre_archivo=dirCodigo.$numero.tgz
tar cvzf $nombre_archivo $directorio??-??-??
rm -r ??-??-??

```

El script cuenta cuantos archivos comprimidos existen en la carpeta */seguridad*, para determinar el número de compresión, los comprime con el nombre *dirCodigo.\$numero.tgz* y borra todos los archivos generados en el mes. Por último vamos a configurar cron para que se ejecute mensualmente y diariamente cada uno de los scripts según corresponda:

```

SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs
# Example of job definition:
# .----- minute (0 - 59)
# | .----- hour (0 - 23)
# | | .----- day of month (1 - 31)
# | | | .----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | |
# * * * * * user-name command to be executed
00 00 1 * * /home/lot94/compresion_mensual.sh
00 00 * * * /home/lot94/copia_diaria.sh

```

Esto quiere decir que *compresion_mensual.sh* se ejecutará a la hora 00:00 del día 1 de cada mes y *copia_diaria.sh* se ejecutará a las 00:00 cada día de la semana.

Cuestión 4

- Pruebe a ejecutar el comando, conectar un dispositivo USB y vuelva a ejecutar el comando. Copie y pegue las líneas que hacen mención al dispositivo conectado (considere usar `dmesg | tail`).
- Explique las diferencias (si las hay) entre o consultar el contenido del archivo `/var/log/dmesg`?

Solución:

- La salida que hace mención a la conexión del dispositivo USB son las siguientes:

```

[ lot94@localhost ~]$ dmesg | tail -25
[ 3859.155240] usb 1-1: new full-speed USB device number 2 using ohci-pci
[ 3859.342210] usb 1-1: New USB device found, idVendor=058f, idProduct=6387
[ 3859.342217] usb 1-1: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 3859.342221] usb 1-1: Product: Mass Storage
[ 3859.342225] usb 1-1: Manufacturer: Generic
[ 3859.342228] usb 1-1: SerialNumber: 3DC5D1E7
[ 3859.675775] usb-storage 1-1:1.0: USB Mass Storage device detected
[ 3859.679591] scsi3 : usb-storage 1-1:1.0
[ 3859.679675] usbcore: registered new interface driver usb-storage
[ 3860.717934] scsi 3:0:0:0: Direct-Access Generic Flash Disk 8.07 PQ: 0 ANSI: 4
[ 3860.724190] sd 3:0:0:0: Attached scsi generic sg2 type 0

```



```
[ 3860.768601] sd 3:0:0:0: [sdb] 15974400 512-byte logical blocks: (8.17 GB/7.61 GiB)
[ 3860.780251] sd 3:0:0:0: [sdb] Write Protect is off
[ 3860.780257] sd 3:0:0:0: [sdb] Mode Sense: 23 00 00 00
[ 3860.790310] sd 3:0:0:0: [sdb] Write cache: disabled, read cache: enabled, doesn't
support DPO or FUA
[ 3860.873104] sdb: sdb1
[ 3860.962896] sd 3:0:0:0: [sdb] Attached SCSI removable disk
[ 3863.743125] FAT-fs (sdb1): Volume was not properly unmounted. Some data may be corrupt.
Please run fsck.
[ 3863.743222] SELinux: initialized (dev sdb1, type vfat), uses genfs_contexts
[ 3917.288285] hrtimer: interrupt took 811643 ns
```

- b) [2] El comando `dmesg` muestra el contenido del registro de mensajes que el sistema ha ido produciendo hasta el momento mientras que el archivo `/var/log/dmesg` contiene los mensajes que se produjeron cuando se produjo el último proceso de arranque.

Cuestión 5

- a) Ejecute el monitor de "System Performance" y muestre el resultado. Incluya capturas de pantalla y comente la información que aparece.
- b) 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.

Solución:

- a) Para ejecutar el monitor de rendimiento hemos utilizado el comando `perfmom` en la consola de windows, como podemos observar tenemos un resumen del sistema que presenta información general del mismo (figura 9):
- **Información del disco físico:** Porcentaje de tiempo inactivo y longitud de cola, para cada disco C,E,E en este caso puesto que se tiene en el sistema un disco reflejado, se muestran los datos anteriores de cada uno de ellos y un a columna total que realiza la media.
 - **Información del procesador:** Muestra el porcentaje de tiempo de interrupción y de procesador además del estado de detención de todos los procesadores del sistema y un total que realiza la media de todos.
 - **Información del interfaz de red:** Muestra el total de bytes de cada adaptador web en el sistema.
 - **Información de la memoria:** Informa de los errores de cache, la memoria libre en ese momento y el porcentaje de bytes en uso.

Además por defecto tenemos el monitor de rendimiento que nos informa del porcentaje de procesador (figura 10), además se nos da la posibilidad de crear nuestros propios recopiladores de datos que pueden realizar una serie de informes con los datos que se especifican a la hora de configuración (figura 9).

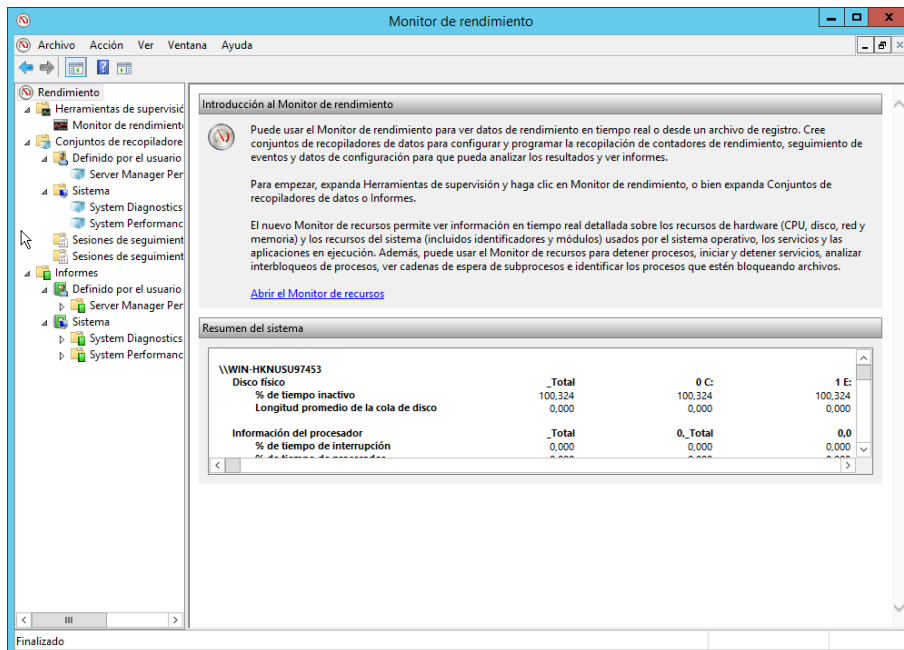


Figura 9: Pantalla de inicio Perfmon.

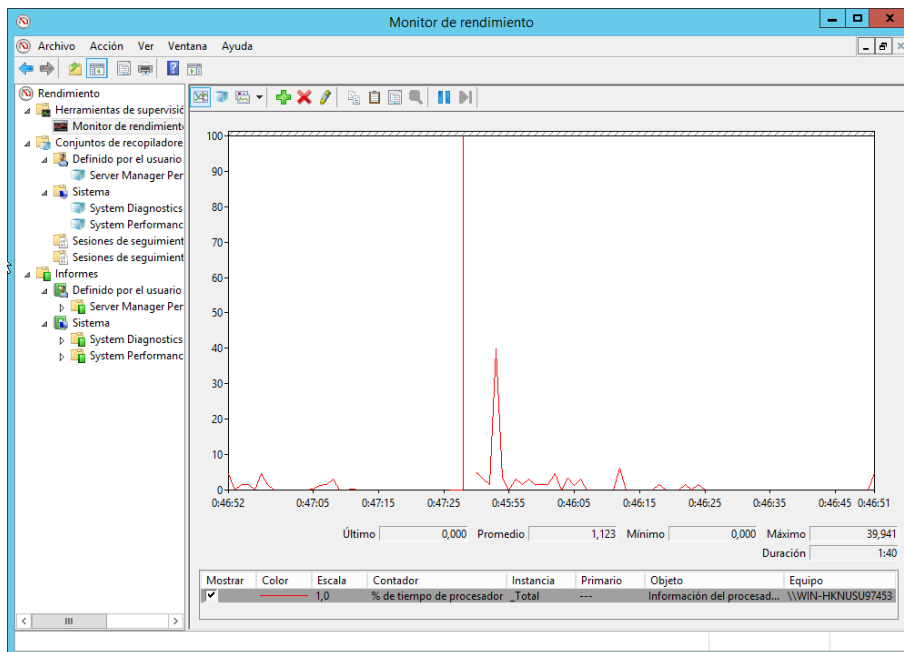


Figura 10: Monitor de rendimiento Perfmon.

- b) Para crear un nuevo conjunto de recopiladores de datos nos dirigimos a los recopiladores definidos por el usuario y seleccionamos la opción Nuevo, obtendremos una ventana (figura 11) en la que podremos poner nombre a nuestro conjunto de recopiladores, seleccionamos crear manualmente y se selecciona contador de rendimiento y datos de seguimiento de eventos (figura 12). Ahora tenemos que seleccionar todos los datos referentes a el procesa-

dor, proceso y servidor web para ello los seleccionamos con todas las instancias del objeto (en mi caso el resultado de elegir _total y <Todas las instancias> fue el mismo) (figura 13), por último confirmamos los contadores de rendimiento (figura 14). Por último seleccionamos la carpeta que guardará los datos y establecemos la carpeta Escritorio/log y finalizamos la creación (figura 15), para que se inicie el recopilador tenemos que iniciarlo en el monitor de rendimiento y se crearían los informes (figura 16) con la información que se pide monitorizar cada 15 segundos (figura 17).

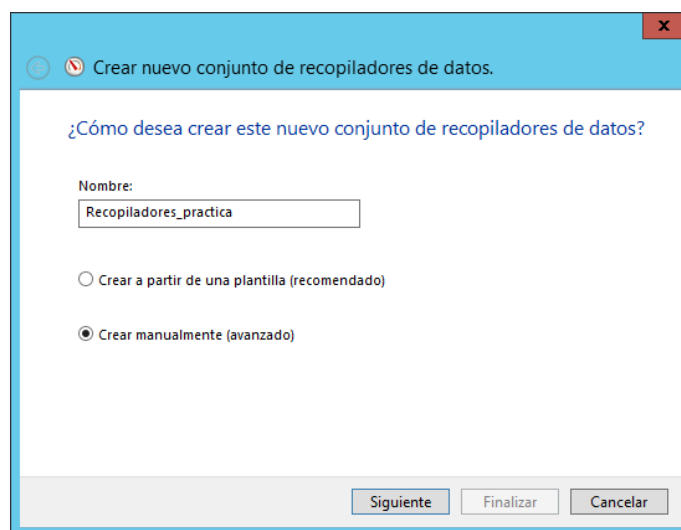


Figura 11: Creación nuevo conjunto de recopiladores.

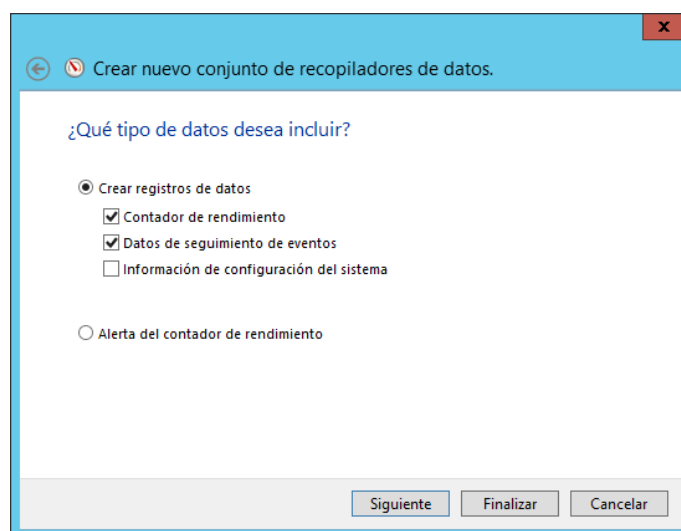


Figura 12: Añadiendo registros de contador de rendimiento y datos de seguimiento de eventos.

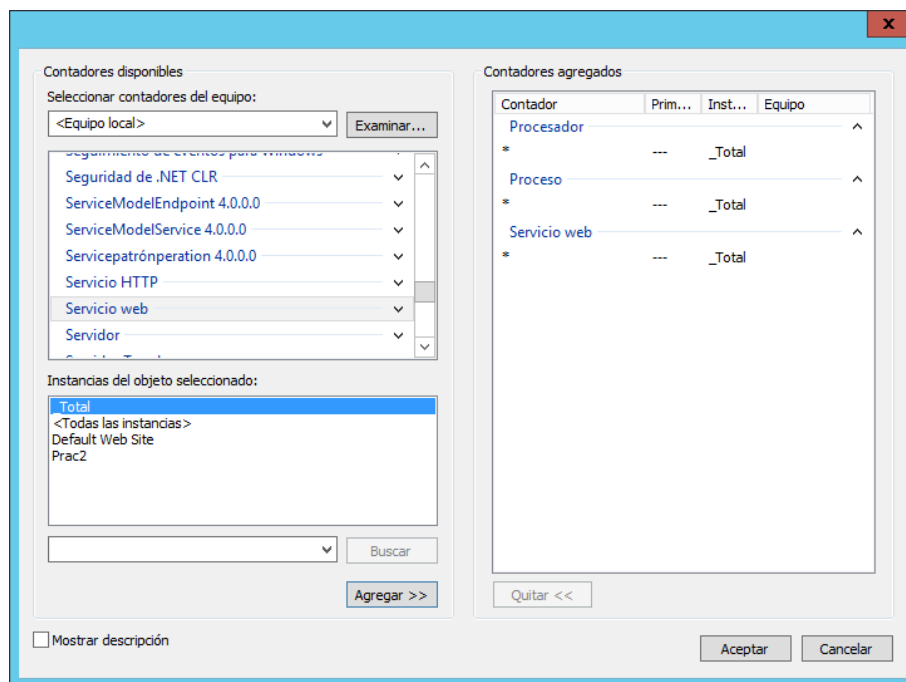


Figura 13: Añadiendo contadores: Procesador, proceso, servicio web.

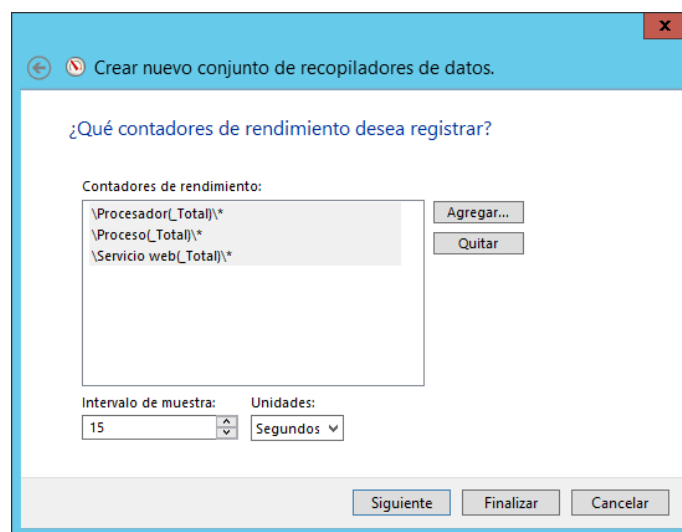


Figura 14: Estableciendo intervalo de muestra.

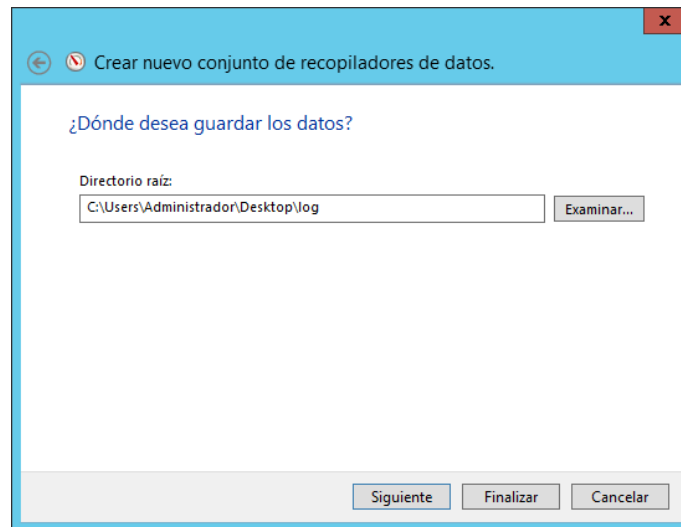


Figura 15: Estableciendo carpeta log.

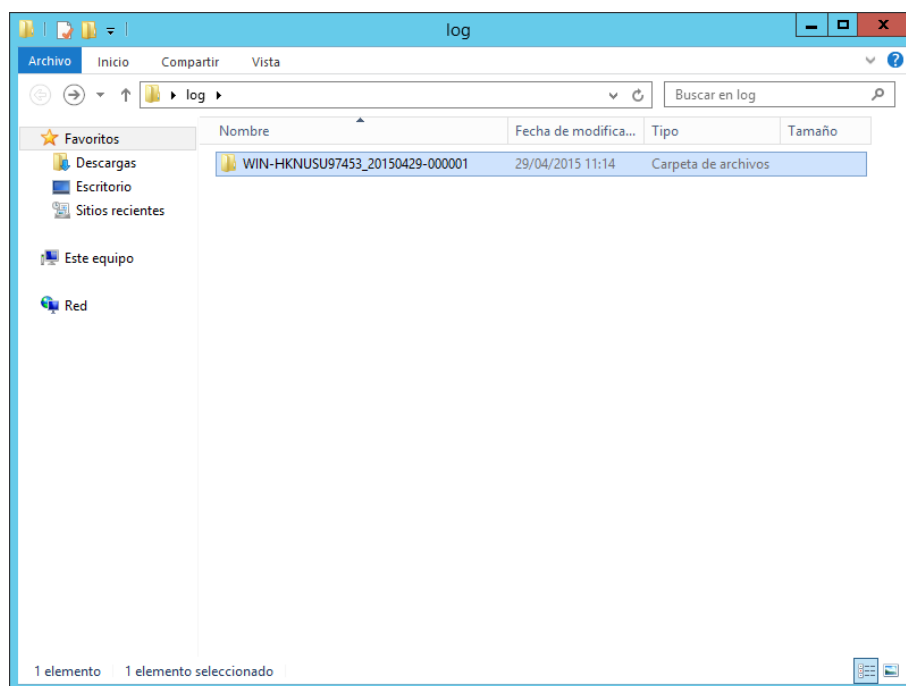


Figura 16: Carpeta log.

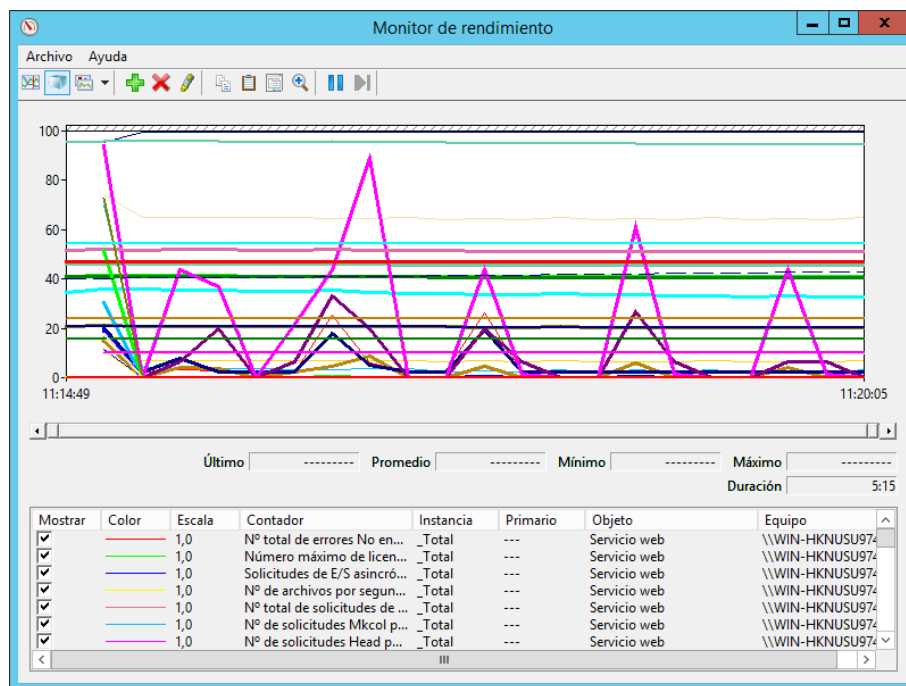


Figura 17: Monitor de rendimiento con los datos cada 15 segundos.

Cuestión 6

Elija uno de los sistemas de monitorización descritos en este apartado e instálelo. Describa los pasos seguidos así como posibles incidencias en la instalación que ha debido resolver. Monitoree uno o varios de sus servidores y presente ejemplos de algunas medidas que considere significativas, explicando su significado y los valores reales observados.

Solución:

El sistema de monitorización elegido es Nagios para instalarlo en CentOS se ha seguido la siguiente guía oficial de instalación [4], en primer lugar hay que tener instalado una serie de paquetes entre ellos todos los que conforman *LAMP* y algunas librerías más por lo que antes de empezar la instalación es necesario ejecutar antes el comando:

```
yum install -y wget httpd php gcc glib glibc-common gd gd-devel make net-snmp
```

Ahora comenzamos la instalación de *Nagios* para ello es necesario descargar *Nagios Core*, además en mi caso instale también *Nagios Plugins*. Ambos pueden obtenerse de la referencia oficial del mismo [10]. Estos pasos permiten la instalación de Nagios en CentOS:

1. **Añadir un usuario u un grupo para Nagios:** Nagios usa un propio usuario para la instalación y posterior configuración del mismo para ello tendremos que ejecutar los comandos:

```
useradd nagios
groupadd nagcmd
usermod -a -G nagcmd nagios
```

2. **Configuración:** Cuando se descarga Nagios de la red se trata de su código fuente por lo que será necesaria una compilación después de la configuración necesaria, después de

descomprimir el archivo usamos el comando `cd` para situarnos en el directorio que contiene los archivos de *Nagios Core* y ejecutamos en primer lugar:

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

En mi caso se produjo un error, al escribir el grupo de usuarios no se realizó de forma correcta y por tanto la configuración no se realizó satisfactoriamente, pero no muestra señales de error hasta que no se avanza en la instalación.

3. **Compilación e instalación:** Una vez configurado pasamos a compilarlo con el comando *make* ahora instalamos nagios junto con una serie de módulos con los comandos:

```
make install
make install-init
make install-config
make install-commandmode
make install-webconf
```

Con esto instalamos todas las configuraciones, en mi caso me di cuenta de error anterior cuando ejecute *make install-commandmode* que obtenía un error de salida. Por último concluimos la instalación con los siguientes comandos:

```
cp -R contrib/eventhandlers/ /usr/local/nagios/libexec/
chown -R nagios:nagios /usr/local/nagios/libexec/eventhandlers
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

Por último iniciamos el servicio de Nagios:

```
/etc/init.d/nagios start
```

4. **Establecer el password del usuario por defecto:** Haciendo uso de la orden:

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

5. **Instalación de plugins:** De manera similar a lo anterior siguiendo los comandos:

```
cd /tmp/nagios-plugins-2.0
./configure --with-nagios-user=nagios --with-nagios-group=nagios
make
make install
```

Para acceder al monitor tenemos que hacerlo a través de su interfaz web a través de la dirección *http://<IP>/nagios* en nuestro caso *http://localhost/nagios* (figura 18).

Con él podremos comprobar la información del servidor en todo momento, desde otra terminal, nos aporta mucha información del sistema, tiene una página que lista los hosts que se encuentran en línea (figura 19) y los servicios de este (figura 20). Ahora está todo correcto pero en caso de que se produzcan incidencias el se indica en la parte superior del mismo dependiendo del host como el que aparece en la figura 20, además también se puede comprobar la información de cada uno de los servicios en todo momento y modificar los chequeos sobre el mismo, y las notificaciones en el sistema.



Figura 18: Nagios instalado funcionando en CentOS.

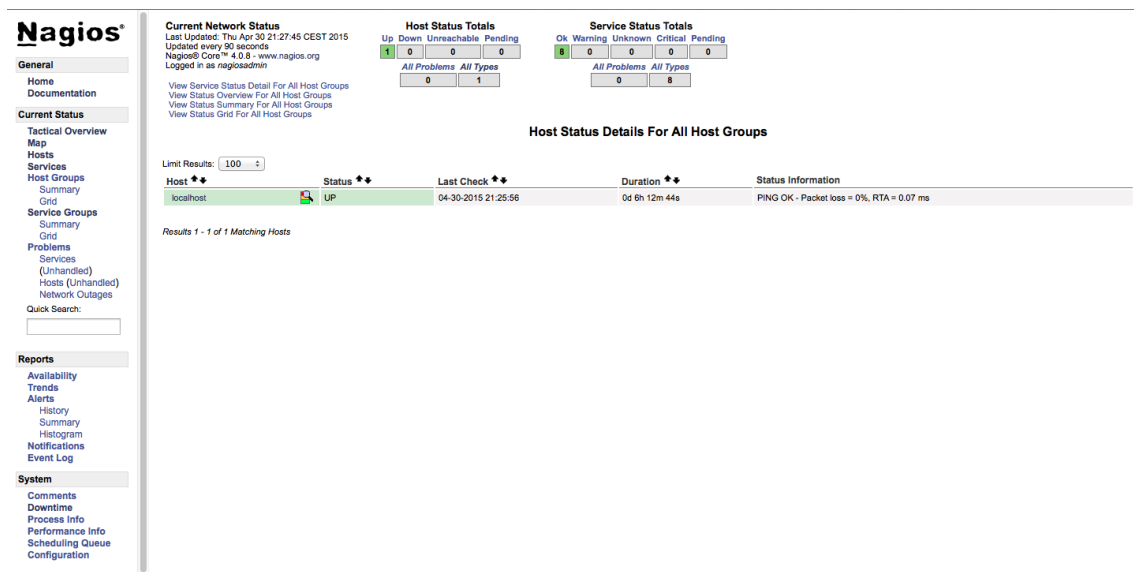


Figura 19: Estado de todos los hosts.

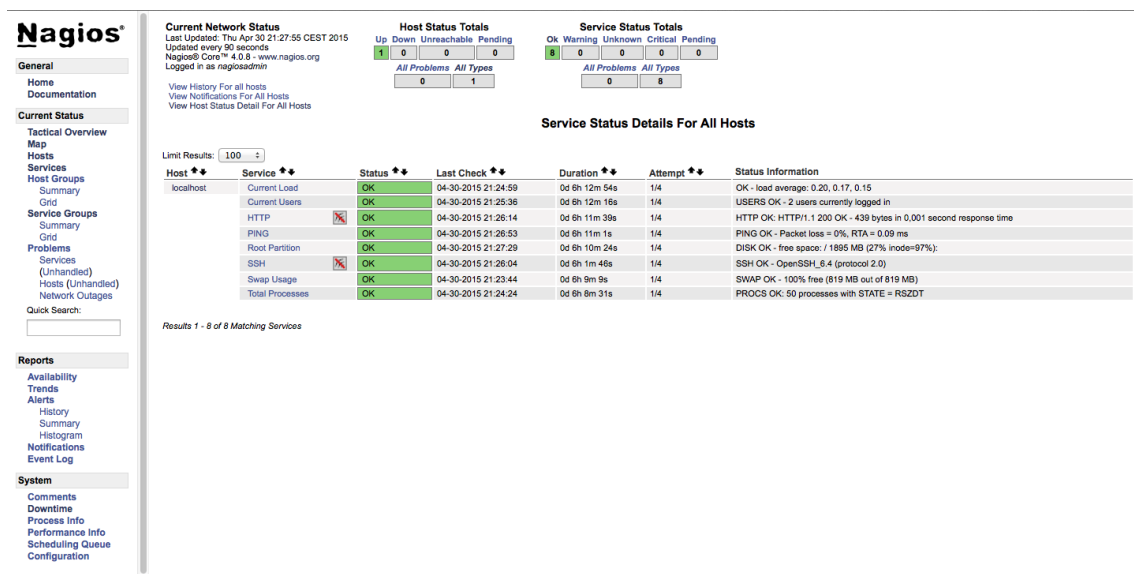


Figura 20: Estados de los servicios del host.

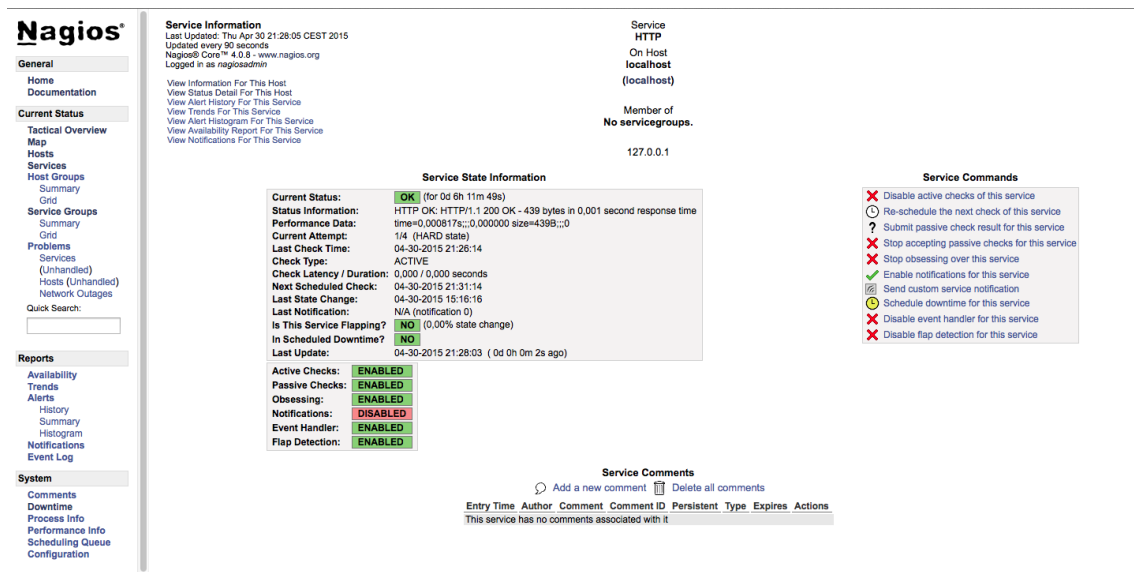


Figura 21: Estado del servicio HTTP.

Nagios es un monitor que da muchas posibilidades debido a que se pueden agregar plugins para ampliar su funcionamiento, por ejemplo en mi caso queremos obtener información de la carga de CPU, Nagios por defecto no tiene un plugin que muestra esta información de forma directa e interpretable por lo que he optado por la instalación de un plugin llamado *pnp4nagios* que da información de los host y servicios del sistema. El proceso de instalación es similar, descargamos *pnp4nagios* de la referencia oficial [9], descomprimos el archivo y ejecutamos:

```
./configure
make all
make install
make install-webconf
```

```
make install-config
make install-init
```

Ahora procedemos a su configuración, en primer lugar nos dirigimos a `/usr/local/nagios/etc/nagios.cfg` y añadimos una serie de líneas:

```
process_performance_data=1
host_perfdata_file=/usr/local/pnp4nagios/var/host-perfdata
service_perfdata_file=/usr/local/pnp4nagios/var/service-perfdata
host_perfdata_file_template=DATATYPE::HOSTPERFDATA\tTIMET::$TIMET$\tHOSTNAME::
$HOSTNAME$\tHOSTPERFDATA::$HOSTPERFDATA$\tHOSTCHECKCOMMAND::$HOSTCHECKCOMMAND$\tHOSTSTATE::
$HOSTSTATE$\tHOSTSTATETYPE::$HOSTSTATETYPE$\tHOSTOUTPUT::$HOSTOUTPUT$
service_perfdata_file_template=DATATYPE::SERVICEPERFDATA\tTIMET::$TIMET$\tHOSTNAME::
$HOSTNAME$\tSERVICEDESC::$SERVICEDESC$\tSERVICEPERFDATA::$SERVICEPERFDATA$\tSERVICECHECKCOMMAND::$SERVICECHECKCOMMAND$\tHOSTSTATE::
$HOSTSTATE$\tHOSTSTATETYPE::$HOSTSTATETYPE$\tSERVICESTATE::$SERVICESTATE$\tSERVICESTATETYPE::$SERVICESTATETYPE$\tSERVICEOUTPUT::$SERVICEOUTPUT$
host_perfdata_file_mode=a
service_perfdata_file_mode=a
host_perfdata_file_processing_interval=15
service_perfdata_file_processing_interval=15
host_perfdata_file_processing_command=process-host-perfdata-file
service_perfdata_file_processing_command=process-service-perfdata-file
```

(El texto a añadir esta simplificado para una uso correcto dirijase a la referencia [5]). Esta es la configuración que va a permitir a pnp4nagios adherirse a nagios y comenzar a realizar los registros de datos (figura 22). Ahora nos dirigimos al archivo `/usr/local/nagios/etc/templates.cfg` para añadir dos templates, esto nos servira para que la interfaz web de nagios entienda que existe información de los servicios y los host y que debe mostrarse:

```
define host {
name          host-pnp
action_url    /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=_HOST_
register      0
}

define service {
name          srv-pnp
action_url    /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=$SERVICEDESC$
register      0
}
```

Por último nos dirigimos al archivo `/usr/local/nagios/etc/localhost.cfg` para terminar la configuración, y ya tan solo tenemos que agregar los templates creados a los servicios y hosts declarados en este archivo, en mi caso lo hice con el servicio PING (figura 25) y la máquina local (figura 23,24,26). Podemos comprobar que *pnp4nagios* funciona accediendo a la dirección `http://localhost/pnp4nagios`, un fallo posterior a la instalación que impedía que se mostrasen los datos proporcionados por el plugin fue que el plugin hace uso de Apache y despues de la instalación para que esta funcione correctamente es necesario reiniciar el servicio de nagios y apache para ello se usa:

```
service httpd restart
service nagios restart
```

Despues de esto ya tendremos *pnp4nagios* operativo.

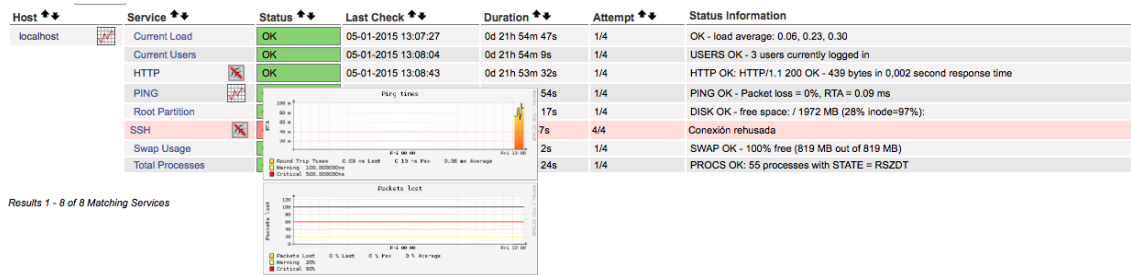


Figura 22: Nagios configurado para que muestre información de pnp4nagios.

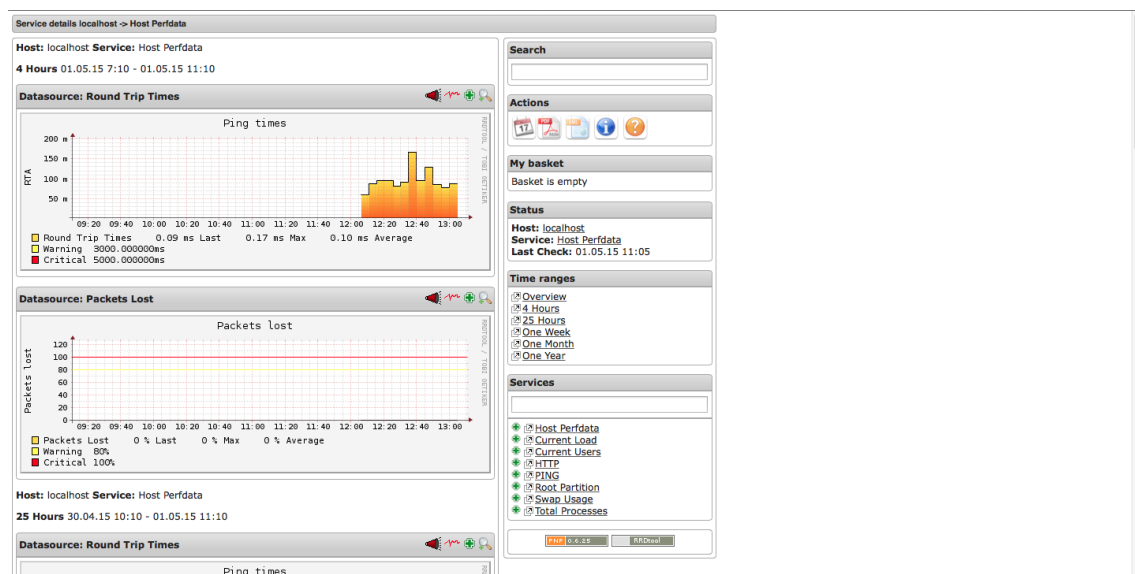


Figura 23: Estado del host, paquetes perdidos.

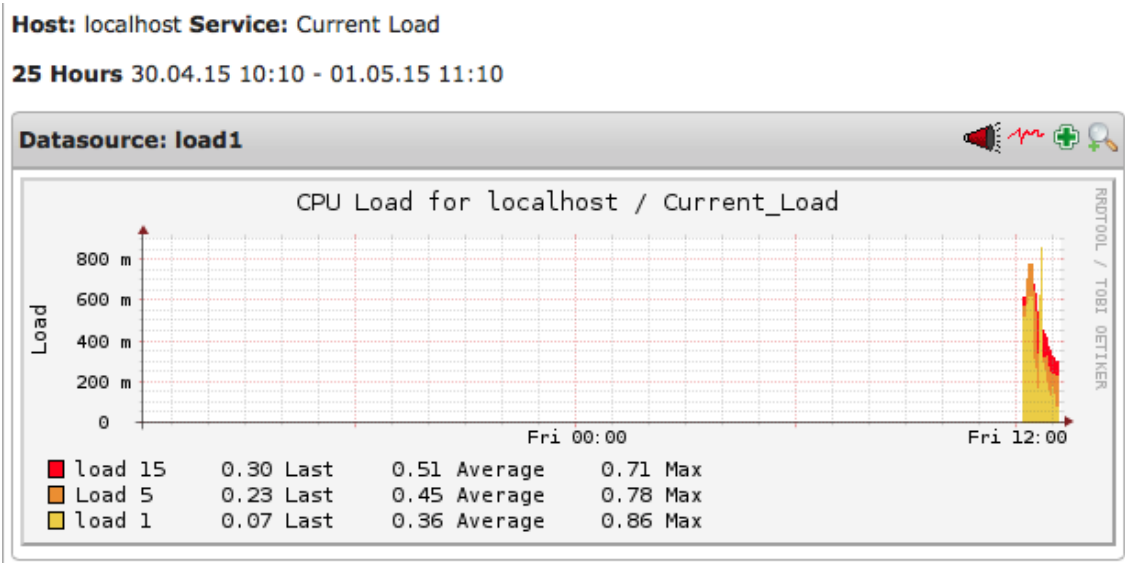


Figura 24: Carga del sistema del Host.



Figura 25: Estado del servicio PING.

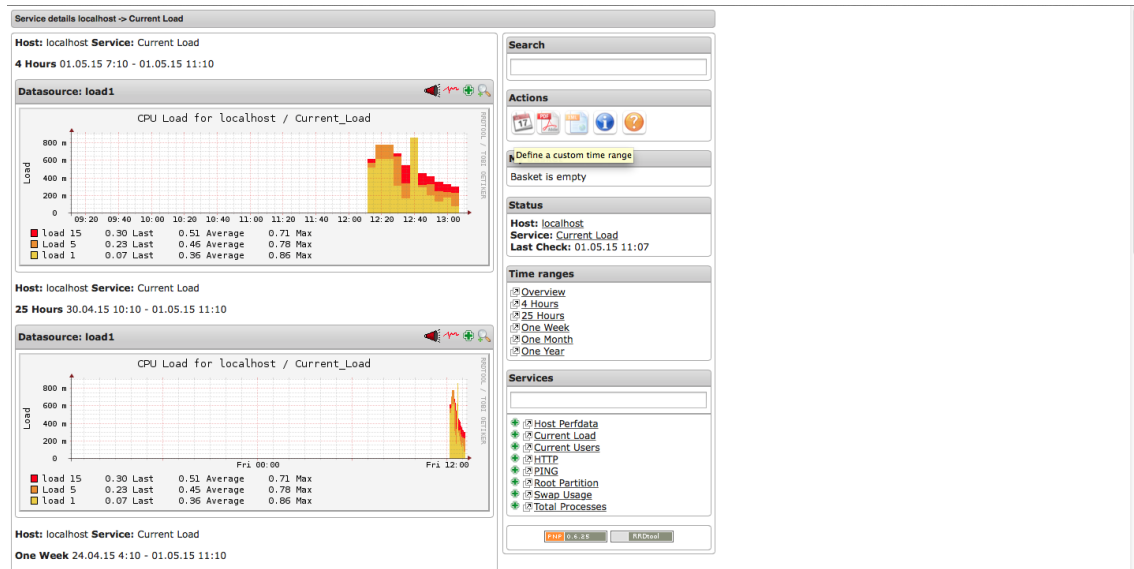


Figura 26: Información extensa sobre la carga del sistema.

Cuestión 7

Diseñe un pequeño modelo de BBDDs para un problema de su elección e impleméntelo en MySQL. También puede emplear un sistema de código abierto del que conozca su diseño de BBDDs. Plantee, una combinación de consultas (al menos dos) que considere significativas y explique los resultados obtenidos en su "profile". Se valorará especialmente, que sea capaz de introducir cambios en el diseño de tablas o en las consultas que mejoren los resultados y que sepa justificar la mejora.

Solución:

El modelo de BBDDs implementado tiene 4 tablas llamadas proveedor, pieza, proyecto y ventas, se trata de un problema en el que cada venta una pieza por parte de un proveedor destinado para un proyecto. El código sql que vamos a usar para generar las tablas es el siguiente:

```
CREATE TABLE proveedor(
codpro char(3) not null,
nompro varchar(30) not null,
status int,
ciudad varchar(15),
primary key (codpro)
);

CREATE TABLE pieza(
codpie char(3) not null,
nompie varchar(10) not null,
color varchar(10),
peso decimal(4),
ciudad varchar(15),
primary key (codpie)
);

CREATE TABLE proyecto(
codpj char(3) not null,
nompj varchar(20) not null,
ciudad varchar(15),
```

```

primary key (codpj)
);
CREATE TABLE ventas(
codpro char(3) references proveedor(codpro),
codpie char(3) references pieza(codpie),
codpj char(3) references proyecto(codpj),
cantidad decimal(4),
primary key (codpro,codpie,codpj)
);

```

Para iniciar el profiler de mysql tenemos que ejecutar la sentencia siguiente:

```

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

```

Figura 27: Activando el profiler.

Ya que por defecto está establecido a 0 y no hace uso del profiler. Ahora debemos seleccionar la base de datos que vamos a usar para ello se creó una BD llamada *Practica*:

```

mysql> create database Practica;
Query OK, 1 row affected (0,00 sec)

```

Figura 28: Creación de la BD Practicas.

```

mysql> use Practica
Database changed

```

Figura 29: Selección base de datos Practicas.

Ejecutamos las sentencias de creación de tablas anteriormente mostrado con el siguiente comando, ya que están almacenadas en un archivo llamado tablas.sql:

```

mysql> \. ~/Escritorio/tablas.sql
Query OK, 0 rows affected (0,08 sec)

Query OK, 0 rows affected (0,11 sec)

Query OK, 0 rows affected (0,20 sec)

Query OK, 0 rows affected (0,07 sec)

```

Figura 30: Creacion de las tablas.

Ahora que hemos creado las tablas vamos a ver que ha ocurrido en el profiler:

```
mysql> mysql> show profiles;
+-----+-----+-----+
+-----+
| Query_ID | Duration | Query
+-----+-----+-----+
+-----+
| 1 | 0.16738200 | CREATE TABLE proveedor(
codpro char(3) not null,
nompro varchar(30) not null,
status int,
ciudad varchar(15),
primary key (codpro)
)
| 2 | 0.09597775 | CREATE TABLE pieza(
codpie char(3) not null,
nompie varchar(10) not null,
color varchar(10),
peso decimal(4),
ciudad varchar(15),
primary key (codpie)
)
| 3 | 0.09999575 | CREATE TABLE proyecto(
codpj char(3) not null,
nompj varchar(20) not null,
ciudad varchar(15),
primary key (codpj)
)
+-----+-----+-----+
4 rows in set, 1 warning (0.00 sec)
```

Figura 31: Estado profiler despues de crear tablas I.

```
+-----+-----+-----+
+-----+
| 4 | 0.02503000 | CREATE TABLE ventas(
codpro char(3) references proveedor(codpro),
codpie char(3) references pieza(codpie),
codpj char(3) references proyecto(codpj),
cantidad decimal(4),
primary key (codpro,codpie,codpj)
)
+-----+-----+-----+
+-----+
4 rows in set, 1 warning (0.00 sec)
```

Figura 32: Estado profiler despues de crear tablas II.

Como podemos darnos cuenta el profiler da información acerca de el orden de introducción de las consultas, la duración de las mismas y cuál fue la consulta, esto lo podemos conocer con el comando "show profiles;", sin embargo si queremos conocer cuanto información acerca de cuanto tardó explícitamente la consulta anterior se usa "show profile [option]". Para probar nuestra base de datos vamos a introducir un conjunto de valores en ella por ejemplo:

```
INSERT INTO proveedor VALUES ('S1','Jose', 2, 'Madrid');
INSERT INTO proveedor VALUES ('S2','Manuel', 1, 'Londres');
INSERT INTO proveedor VALUES ('S3','Luisa', 3, 'Lisboa');
INSERT INTO proveedor VALUES ('S4','Pedro', 4, 'Paris');
INSERT INTO proveedor VALUES ('S5','Maria', 5, 'Roma');

INSERT INTO pieza VALUES ('P1','Tuerca', 'Gris', 1, 'Madrid');
INSERT INTO pieza VALUES ('P2','Tornillo', 'Rojo', 1.5, 'Paris');
INSERT INTO pieza VALUES ('P3','Arandela', 'Blanco', 4, 'Lisboa');

INSERT INTO proyecto VALUES ('J1', 'proyecto 1', 'Londres');
INSERT INTO proyecto VALUES ('J2', 'proyecto 2', 'Londres');
INSERT INTO proyecto VALUES ('J3', 'proyecto 3', 'Paris');
INSERT INTO proyecto VALUES ('J4', 'proyecto 4', 'Roma');

INSERT INTO ventas VALUES ( 'S1','P1','J1',120);
INSERT INTO ventas VALUES ( 'S1','P1','J2',100);
```

```
INSERT INTO ventas VALUES ( 'S3','P2','J3',220);
INSERT INTO ventas VALUES ( 'S5','P3','J1',300);
```

Como podemos ver se puede ver cuanto ha tardado cada una de esas consultas:

```
mysql> show profiles;
+-----+-----+-----+
| Query_ID | Duration | Query |
+-----+-----+-----+
| 11 | 0.00160075 | INSERT INTO proveedor VALUES ('S5','Maria', 5, 'Roma') |
| 12 | 0.00186375 | INSERT INTO pieza VALUES ('P1','Tuerca', 'Gris', 1, 'Madrid') |
| 13 | 0.00389150 | INSERT INTO pieza VALUES ('P2','Tornillo', 'Rojo', 1.5, 'Paris') |
| 14 | 0.00541075 | INSERT INTO pieza VALUES ('P3','Arandela', 'Blanco', 4, 'Lisboa') |
| 15 | 0.00220800 | INSERT INTO proyecto VALUES ('J1', 'proyecto 1', 'Londres') |
| 16 | 0.00413275 | INSERT INTO proyecto VALUES ('J2', 'proyecto 2', 'Londres') |
| 17 | 0.02543400 | INSERT INTO proyecto VALUES ('J3', 'proyecto 3', 'Paris') |
| 18 | 0.01914275 | INSERT INTO proyecto VALUES ('J4', 'proyecto 4', 'Roma') |
| 19 | 0.00269125 | INSERT INTO ventas VALUES ( 'S1','P1','J1',120) |
| 20 | 0.00280650 | INSERT INTO ventas VALUES ( 'S1','P1','J2',100) |
| 21 | 0.00471475 | INSERT INTO ventas VALUES ( 'S3','P2','J3',220) |
| 22 | 0.00311225 | INSERT INTO ventas VALUES ( 'S5','P3','J1',300) |
| 23 | 0.00025300 | select * from proveedor |
| 24 | 0.00024775 | select * from pieza |
| 25 | 0.00023175 | select * from ventas |
+-----+-----+-----+
15 rows in set, 1 warning (0.00 sec)
```

Figura 33: Estado profiler despues de insertar en las tablas.

Ahora vamos a proceder a comparar una serie de consultas utilizando el profiler:

- **Creación y alteración de tablas:** Normalmente una alteración de una tabla, es más costosa que una creación, esto es debido a que la alteración tiene que recrear la tabla, como podemos ver en las figuras 34-35, la creación de tablas en primer lugar tiene que ejecutar menos procesos y el más costoso de ellos es crear la propia tabla sin embargo la alteración además de tener que completar dicho proceso de creación tiene que añadir la nueva información que se le ha dado por ello que la alteración es más costosa como podemos comprobar en la figura 36 en las entradas 26 y 27.

```
mysql> create table Contactos (ID int(3) primary key, nombre varchar(20), telefono int(10));
Query OK, 0 rows affected (0.06 sec)

mysql> show profile cpu;
+-----+-----+-----+-----+
| Status | Duration | CPU_user | CPU_system |
+-----+-----+-----+-----+
| starting | 0.000073 | 0.000011 | 0.000057 |
| checking permissions | 0.000010 | 0.000002 | 0.000008 |
| opening tables | 0.000151 | 0.000024 | 0.000127 |
| creating table | 0.060478 | 0.000000 | 0.003328 |
| After create | 0.000000 | 0.000000 | 0.000000 |
| query end | 0.000000 | 0.000000 | 0.000000 |
| closing tables | 0.000000 | 0.000000 | 0.000000 |
| freeing items | 0.000014 | 0.000000 | 0.000000 |
| cleaning up | 0.000013 | 0.000000 | 0.000000 |
+-----+-----+-----+-----+
9 rows in set, 1 warning (0.00 sec)
```

Figura 34: Tiempos de creación de una tabla.


```
mysql> alter table Contactos add(direccion varchar(30));
Query OK, 0 rows affected (0,13 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> show profile cpu;
```

Status	Duration	CPU_user	CPU_system
starting	0.000059	0.000000	0.000000
checking permissions	0.000006	0.000000	0.000000
checking permissions	0.000014	0.000000	0.000000
init	0.000003	0.000000	0.000000
Opening tables	0.000388	0.000000	0.000393
setup	0.000031	0.000000	0.000000
creating table	0.053918	0.000000	0.000470
After create	0.000215	0.003110	0.000000
System lock	0.000008	0.000000	0.000000
preparing for alter table	0.057889	0.000000	0.000000
altering table	0.000962	0.000000	0.000000
committing alter table to stor	0.013368	0.005938	0.000000
end	0.000011	0.000000	0.000000
query end	0.000008	0.000000	0.000000
closing tables	0.000005	0.000000	0.000000
freeing items	0.000016	0.000000	0.000000
cleaning up	0.000010	0.000000	0.000000

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

Figura 35: Tiempos de alteración de una tabla.

```
mysql> show profiles;
```

Query_ID	Duration	Query
14	0.00541075	INSERT INTO pieza VALUES ('P3','Arandela', 'Blanco', 4, 'Lisboa')
15	0.00220800	INSERT INTO proyecto VALUES ('J1', 'proyecto 1', 'Londres')
16	0.00413275	INSERT INTO proyecto VALUES ('J2', 'proyecto 2', 'Londres')
17	0.02543400	INSERT INTO proyecto VALUES ('J3', 'proyecto 3', 'Paris')
18	0.01914275	INSERT INTO proyecto VALUES ('J4', 'proyecto 4', 'Roma')
19	0.00269125	INSERT INTO ventas VALUES ('S1','P1','J1',120)
20	0.00280650	INSERT INTO ventas VALUES ('S1','P1','J2',100)
21	0.00471475	INSERT INTO ventas VALUES ('S3','P2','J3',220)
22	0.00311225	INSERT INTO ventas VALUES ('S5','P3','J1',300)
23	0.00025300	select * from proveedor
24	0.00024775	select * from pieza
25	0.00023175	select * from ventas
26	0.06073725	create table Contactos (ID int(3) primary key, nombre varchar(20), telefono int(10))
27	0.12691075	alter table Contactos add(direccion varchar(30))
28	0.00009875	show profiles cpu

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

Figura 36: Tiempo total de creación y alteración de una tabla.

- **Comparación de consultas con distintas construcciones:** Se han preparado dos consultas sobre la base de datos con el mismo resultado pero con construcciones distintas, la primera de ellas (figura 37) usa un operador *join*, mientras que la segunda no usa ningún operador (figura 39) como podemos comprobar el resultado es el mismo pero fijandonos en los tiempos aportados por el profiler (figura 38-40) la consulta que usa el operador *join* hace mucho más uso de la CPU que la que no lo usa aunque la duración sea prácticamente

la misma (figura 41), no es una gran carga para la CPU sin embargo conviene cargarla lo menos posible y hacer uso de consultas optimas para ello.

```
mysql> select a.ciudad from proveedor a join pieza b on a.ciudad = b.ciudad where b.codpie='P1';
+-----+
| ciudad |
+-----+
| Madrid |
+-----+
1 row in set (0,00 sec)
```

Figura 37: Consulta I.

```
mysql> show profile cpu;
+-----+-----+-----+-----+
| Status          | Duration | CPU_user | CPU_system |
+-----+-----+-----+-----+
| starting        | 0.000079 | 0.000012 | 0.000059 |
| checking permissions | 0.000006 | 0.000001 | 0.000004 |
| checking permissions | 0.000005 | 0.000001 | 0.000003 |
| Opening tables  | 0.000031 | 0.000005 | 0.000023 |
| init            | 0.000034 | 0.000006 | 0.000026 |
| System lock     | 0.000012 | 0.000001 | 0.000009 |
| optimizing      | 0.000021 | 0.000004 | 0.000016 |
| statistics      | 0.000069 | 0.000011 | 0.000053 |
| preparing       | 0.000019 | 0.000003 | 0.000014 |
| executing       | 0.000004 | 0.000001 | 0.000002 |
| Sending data    | 0.000030 | 0.000005 | 0.000023 |
| end             | 0.000006 | 0.000000 | 0.000004 |
| query end       | 0.000007 | 0.000001 | 0.000005 |
| closing tables  | 0.000010 | 0.000002 | 0.000007 |
| freeing items   | 0.000032 | 0.000005 | 0.000025 |
| cleaning up     | 0.000014 | 0.000002 | 0.000010 |
+-----+-----+-----+-----+
16 rows in set, 1 warning (0,00 sec)
```

Figura 38: Tiempos de la consulta I.

```
mysql> select a.ciudad from proveedor a, pieza b where b.codpie='P1' and a.ciudad = b.ciudad;
+-----+
| ciudad |
+-----+
| Madrid |
+-----+
1 row in set (0,00 sec)
```

Figura 39: Consulta II.

```
mysql> show profile cpu;
```

Status	Duration	CPU_user	CPU_system
starting	0.000077	0.000000	0.000000
checking permissions	0.000007	0.000000	0.000000
checking permissions	0.000005	0.000000	0.000000
Opening tables	0.000024	0.000000	0.000000
init	0.000034	0.000000	0.000000
System lock	0.000011	0.000000	0.000000
optimizing	0.000017	0.000000	0.000000
statistics	0.000094	0.000000	0.000202
preparing	0.000020	0.000000	0.000000
executing	0.000003	0.000000	0.000000
Sending data	0.000030	0.000000	0.000000
end	0.000006	0.000000	0.000000
query end	0.000006	0.000000	0.000000
closing tables	0.000010	0.000000	0.000000
freeing items	0.000014	0.000000	0.000000
cleaning up	0.000012	0.000000	0.000000

```
16 rows in set, 1 warning (0.00 sec)
```

Figura 40: Tiempos de la consulta II.

```
60 | 0.00037400 | select a.ciudad from proveedor a join pieza b on a.ciudad = b.ciudad where b.codpie='P1'
```

```
61 | 0.00036750 | select a.ciudad from proveedor a, pieza b where b.codpie='P1' and a.ciudad = b.ciudad
```

Figura 41: Tiempo total de las consultas.

Referencias

1. <http://manpages.ubuntu.com/manpages/jaunty/man8/logrotate.8.html>
2. <http://en.wikipedia.org/wiki/Dmesg>
3. <http://systemadmin.es/2011/02/anadir-o-quitar-discos-en-fallo-de-un-raid-por-software>
4. http://assets.nagios.com/downloads/nagioscore/docs/Installing_Nagios_Core_From_Source.pdf
5. <http://bitflip.net/graphing-nagios-services-with-pnp4nagios-0-6-41/>
6. <http://dev.mysql.com/doc/refman/5.0/en/show-profile.html>
7. <http://linux.die.net/man/8/mdadm>
8. https://www.howtoforge.com/replacing_hard_disks_in_a_raid1_array
9. <https://docs.pnp4nagios.org>
10. <http://www.nagios.org/download>