# INGENIERÍA DE SERVIDORES

## Guión de la Práctica 3

---

## Monitorización de servicios

---

## Curso 2015-2016

# GUIÓN DE PRÁCTICAS
# Práctica 3

## Índice



## Índice de Figuras

# GUIÓN DE PRÁCTICAS
# Práctica 3

**TÍTULO DE LA PRÁCTICA:**

Monitorización de servicios

Centro: Escuela Técnica Superior de Ingeniería Informática y Telecomunicación

Asignatura: Ingeniería de Servidores

**Profesores:** Alberto Guillén Perales (autor)
Gonzalo Ruiz García

Grupo: A

## 1.- OBJETIVOS

Los objetivos mínimos que el alumno debe alcanzar son:

1) Conocer y saber usar las herramientas que permitan obtener datos sobre el sistema a nivel hardware y software (SO y servicios).
2) Saber interpretar los resultados proporcionados por las aplicaciones de monitorización.
3) Conocer los archivos que proporcionan información del sistema.
4) Saber utilizar "profilers" para analizar el código de las aplicaciones y scripts programados.

# 2.- ENUNCIADO DE LA PRÁCTICA

Esta práctica consiste en la utilización de herramientas de monitorización del sistema para visualizar cómo ejercita el sistema ciertas actividades que los usuarios u otros servicios generan. En clases de teoría verán el concepto de carga, de modo que ya sabrán cómo "monitorizarla", es decir, ver cómo afecta ésta al sistema real.

Las herramientas de monitorización presentan medidas del sistema en tiempo real, permitiendo generar informes e históricos que puedan ser de utilidad para un análisis a posteriori (*offline*).

El objetivo de esta práctica es monitorizar varios sistemas mientras realizan ciertas tareas. Posteriormente, se deben **resumir y representar los resultados de un modo adecuado**.

Se da por hecho que si no tiene algún servicio instalado (p.ej. Intérprete de python) usted ya tiene el conocimiento para instalarlo. **En tales casos, indique cómo lo ha hecho.**

# 3.- MONITORIZACIÓN DE SISTEMAS LINUX

Hay una serie de comandos que permiten ver el estado de la máquina y muestran el resultado en la consola. A continuación veremos algunos de ellos y el alumno, con la ayuda de las **páginas del manual** y las referencias proporcionadas en la presentación de la asignatura, deberá interpretar los resultados.

## 3.1.- CONOCIENDO EL SUBSISTEMA DE ARCHIVOS

En Linux (UNIX) todo se manipula a través de archivos de una manera cómoda y transparente. Existe un directorio especial: /proc (visto en clase de teoría) y /var (algo se ha comentado también al respecto).

**Cuestión 1:** 5.a) ¿Qué archivo le permite ver qué programas se han instalado con el gestor de paquetes? 5.b) ¿Qué significan las terminaciones .1.gz o .2.gz de los archivos en ese directorio?

Con el programa ***gnome-system-log*** podemos tener acceso a todos estos archivos de una manera cómoda a través de una GUI.

### 3.2.1 Volviendo con el RAID1

Dentro de /proc existe un archivo para el estado de los *multidevice* como es el caso de nuestro /dev/md0 creado en la P1.

Vamos a monitorizar el proceso de sustituir un disco dañado por uno nuevo y cómo podemos saber cuándo el sistema está preparado para continuar operando con normalidad.

Con la máquina encendida, quite uno de los dos discos y luego pase a insertarle/crear uno nuevo (que estará completamente vacío).

Con el comando "mdadm" puede eliminar del RAID el disco defectuoso y añadir el nuevo dispositivo.

Un modo de *monitorizar* el proceso de réplica mediante:

```
watch -n2 cat  /proc/mdstat
```

**Cuestión opcional 1**: Indique qué comandos ha utilizado para realizarlo así como capturas de pantalla del proceso de reconstrucción del RAID.

## 3.2.- PROGRAMACIÓN DE TAREAS CON CRON

El comando cron permite ejecutar cada cierto intervalo de tiempo una tarea concreta. Esto es muy útil de cara a recopilar información o monitorizar el sistema realizando una tarea determinada, por ejemplo, enviar un correo electrónico cuando la carga esté por encima de un valor determinado.

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

## 3.3.- ANALIZANDO QUÉ OCURRE EN EL KERNEL CON DMESG

El *kernel* de Linux permite conocer qué actividad ha ocurrido gracias a los mensajes que proporciona el kernel. Esto es especialmente útil para detectar problemas con el HW o periféricos.

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

## 3.4.- MONITOR GENERAL GNOME-SYSTEM-MONITOR

Además de los comandos comentados previamente, existe una aplicación que permite monitorizar tanto los procesos como los sistemas de memoria y de red: gnome-system-monitor.

# 4.- MONITORIZANDO WINDOWS: PERFMON

En Windows, el equivalente al programa top es el "Administrador de tareas" que puede invocarse tras pulsar Ctrl+Alt+Supr.

Sin embargo, existe una aplicación mucho más potente que nos permite controlar el estado del servidor de manera visual así como recopilar datos para su posterior uso. Su nombre es perfmon. Esta herramienta se encarga de recopilar los datos proporcionados por otros sistemas dentro de Windows.

Para iniciar perfmon, podemos abrir una consola y ejecutar el comando con el mismo nombre (para una lista con los comandos disponibles, visite: http://technet.microsoft.com/en-us/library/cc772390%28v=ws.10%29.aspx)

Las características detalladas del software se pueden consultar en http://technet.microsoft.com/en-us/library/cc749249.aspx?ppud=4 aunque en el mismo programa, haciendo click en "Obtener más información" también obtendrá la documentación correspondiente.

Nos centraremos en la creación de un recopilador de datos y en la realización de un informe de diagnóstico.

Tras ejecutar el monitor, en el panel de la izquierda encontraremos "Conjunto de recopiladores de datos" y dentro de la subsección veremos unos definidos por el usuario (debe estar vacío) y otros definidos por el sistema. Pulsando sobre uno de éstos con el botón derecho, podemos iniciarlos (puesto que se encuentran detenidos por defecto) y, tras esperar un minuto, se generará automáticamente un informe.

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

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

- Todos los referentes al procesador, al proceso y al servicio web.
- Intervalo de muestra 15 segundos
- Almacene el resultado en el directorio Escritorio\logs

Incluya las capturas de pantalla de cada paso.

# 5.- MONITORIZANDO EL HARDWARE

Además del estado del software del servidor, también existen programas que nos permiten ver el estado del hardware de nuestra máquina. En primer lugar, muchas BIOS nos permiten acceder a cierta información sobre el estado del HW; sin embargo, para no tener que reiniciar, podemos utilizar estas herramientas:

*Para Linux:*

- Para la temperatura del HD tenemos: hddtemp
- Proyecto lm-sensors: http://lm-sensors.org/
- que posee una GUI: xsensors

*Para Windows:*

- Open Hardware Monitor: http://openhardwaremonitor.org/ (también funciona bajo Linux aunque depende de Mono: http://www.mono-project.com/Main_Page )
- SpeedFan: http://www.almico.com/speedfan.php
- RealTemp: http://www.techpowerup.com/realtemp/
- Core Temp: www.alcpu.com/CoreTemp/
- CPUID: http://www.cpuid.com/softwares/hwmonitor.html
- Speccy: http://www.piriform.com/speccy

**Cuestión 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.

# 6.- OTROS MONITORES DEL SISTEMA

Además de los comandos integrados vistos en teoría y lo que ha manejado en prácticas, existen otros programas muy populares que permiten monitorizar el sistema.

Hay algunos de entorno corporativo de grandes empresas como **NetApp (**http://www.netapp.com/es/products/management-software/), aunque los que se verán en las siguientes subsecciones también son usados por grandes instituciones y empresas.

## 6.1.- MUNIN

Munin está disponible en http://munin-monitoring.org/.

Para su instalación (en CentOS) puede hacerlo compilando el código fuente (como ha podido hacer con lm_sensors) alojado en la página o a través del paquete disponible en el repositorio EPEL (http://fedoraproject.org/wiki/EPEL/es). "¿Cómo puedo utilizar estos paquetes adicionales?" es el título de la subsección dentro de la página donde se explica cómo activar el repositorio que contiene los paquetes. Tan solo debe instalar un .rpm y podrá usar yum para instalar munin.

Para Ubuntu, está disponible sin tener que añadir ningún repositorio adicional.

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

## 6.2.- NAGIOS

Es otro software muy usado para monitorizar sistemas. En su página web http://www.nagios.org/ se encuentra más información así como diversos tutoriales sobre cómo instalarlo en CentOS.

**Cuestión opcional 2:** instale Nagios en su sistema (el que prefiera) documentando el proceso y muestre el resultado de la monitorización de su sistema comentando qué aparece.

## 6.3.- GANGLIA

Es un proyecto alojado en http://ganglia.sourceforge.net/ que monitoriza sistemas de cómputo distribuidos (normalmente para altas prestaciones) y permite una gran escalabilidad.

Como puede verse en su página, importantes instituciones y compañías lo usan (p.ej. UC Berkely, MIT, Twitter, …).

**Cuestión opcional 3:** Haga lo mismo que con Munin.

## 6.4.- ZABBIX

Es otro programa que permite monitorizar el sistema también de código abierto. Su instalación es relativamente sencilla ya que solo hay que añadir los repositorios (visto en la práctica anterior) e instalarlo con el gestor de paquetes

**Cuestión opcional 4:** Prueba a instalar este monitor es alguno de sus tres sistemas. Realice capturas de pantalla del proceso de instalación y comente capturas de pantalla del programa en ejecución.

## 6.5.- CACTI

Cacti (http://www.cacti.net/) es un *front-end* para **RRDtool** (http://oss.oetiker.ch/rrdtool/) que permite monitorizar muchos parámetros sin perjudicar mucho el rendimiento del sistema.

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

## 6.6.- AWSTATS

Es un monitor de servicios específicos de servidores web, p.ej.: http, ftp, correo. La página del proyecto es http://awstats.sourceforge.net/. Se puede compilar el código fuente aunque están disponibles los paquetes en los repositorios.

**Cuestión opcional 6:** Instale el monitor y muestre y comente algunas capturas de pantalla.

## 6.7 MONITORIZANDO UN SERVICIO (o ejecución de un programa)

Hay un conjunto de programas que permiten hacer un seguimiento de las llamadas al sistema realizadas por un programa (o servicio) en ejecución, p.ej., strace (system call tracer).

Este tipo de programas pueden ser útiles de cara a detectar problemas que no se muestran en los archivos de "log".

En las siguientes direcciones tiene ejemplos de uso para estos programas, y podrá apreciar su utilidad:

- http://chadfowler.com/blog/2014/01/26/the-magic-of-strace/.
- https://www.debian-administration.org/article/352/Using_strace_to_debug_application_errors.
- http://blog.softlayer.com/2013/sysadmin-tips-and-tricks-using-strace-to-monitor-system-

calls#utm_source=twitter&utm_medium=social&utm_content=beyond-the-command-line-with-strace&utm_campaign=blog_development-tips-and-tricks

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

# 7.- PROFILING

"Profiling" podría considerarse como una monitorización estática en el sentido de que no estamos controlando el comportamiento de un sistema en un momento concreto sino la ejecución de un programa/script/consulta/operación en el momento en el que lo deseemos.

Es un matiz suave que en lengua inglesa y dentro del ámbito tecnológico (el término profiler no está recogido en el diccionario Cambridge) hace que los términos "monitor" y "profiling" sean distintos.

En este apartado vamos a ver algunos programas que permiten analizar el comportamiento de un elemento de ejecución concreto de modo que podamos ver qué parte es menos eficiente y dónde se puede mejorar el rendimiento.

## 7.1 EJECUCIÓN DE PROGRAMAS

### 7.1.1 GPROF Y VALGRIND

Podemos ver cómo se ejecuta un programa y ver cuánto tiempo pasa en cada función con la utilidad gprof.

La documentación detallada se encuentra en: https://sourceware.org/binutils/docs/gprof/ (página mantenida por Red Hat), donde hay un tutorial con ejemplos de uso muy detallado y bien descrito.

Otro programa muy bueno, interesante y popular de cara al "profiling" de un programa compilado es Valgrind: http://valgrind.org/, que posee GUIs, permite seguir las hebras, etc.

Por cuestión de tiempo, no podemos detenernos en probar estos programas. Sin embargo debe saber que con estos programas podríamos analizar el comportamiento de páginas web escritas en C++ (usando, p.ej., http://www.webtoolkit.eu/wt y http://cppcms.com/wikipp/en/page/main )

**Cuestión opcional 7**: Desarrolle una página en C o C++ y analice su comportamiento usando valgrind. Visite http://www.cs.tut.fi/~jkorpela/forms/cgic.html para ver un ejemplo sencillo de una página web generada por un programa escrito en C.

## 7.1.2 PHP

Para estudiar el comportamiento de los scripts que desarrollemos podemos usar varios profilers:

Xdebug: http://www.xdebug.org/index.php
XHProf:http://pecl.php.net/package/xhprof https://github.com/facebook/xhprof

Además de la documentación oficial, en la página de un prestigioso sistema de enseñanza virtual hay unos tutoriales interesantes sobre cómo realizar el profiling. http://docs.moodle.org/dev/Profiling_PHP

**Cuestión opcional 8**: Desarrolle un script en PHP y analice su ejecución con alguno (o los dos) profilers.

## 7.1.3 PYTHON

Para hacer "profiling" de los scripts escritos en este lenguaje se puede utilizar:
http://docs.python.org/2/library/profile.html

**Cuestión opcional 9**: Escriba un script en python y analice su comportamiento usando el profiler presentado.

## 7.1.4 POWERSHELL

Al igual que con los otros lenguajes de script, PowerSHELL posee un "profiler": PoshProfiler

http://gallery.technet.microsoft.com/Powershell-script-profiler-4382ffad

**Cuestión opcional 10**: Escriba un script en PowerShell y analice su comportamiento usando el profiler presentado.

## 7.2 BASES DE DATOS

### 7.2.1 MySQL

El mismo MySQL que instalamos en la práctica anterior tiene un "profiler" para analizar cuánto tardan las consultas.

Puede ver la documentación en:

http://dev.mysql.com/doc/refman/5.0/en/show-profiles.html
http://dev.mysql.com/doc/refman/5.0/en/show-profile.html

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

### 7.2.2 MongoDB

Al igual que MySQL, MongoDB también trae integrado un "profiler": http://docs.mongodb.org/manual/tutorial/manage-the-database-profiler/

**Cuestión opcional 11**: Al igual que ha realizado el "profiling" con MySQL, realice lo mismo con MongoDB y compare los resultados (use la misma información y la misma consulta, hay traductores de consultas SQL a Mongo).

# 8.- NORMATIVA

La misma que la especificada en la guía de prácticas disponible en SWAD.

**Para que la práctica sea apta, debe contestar a todas las cuestiones y, como mínimo, a dos de las opcionales.**