

# Administración de Servidores

Planificación de capacidad  
Medición de uso de recursos y  
resolución de problemas

Creado por

Iñaki Fernández de Viana y González

Huelva, octubre 2020

# Sobre Nosotros

## Iñaki Fernández de Viana y González



Despacho 128. Escuela Técnica  
Superior de Ingeniería



Dpto. De Tecnologías de la Información



[i.fviana@dti.uhu.es](mailto:i.fviana@dti.uhu.es)



+34 959217378



# Objetivos

- **Peso:** 6
- **Descripción:** Los candidatos deben ser capaces de medir recursos hardware y anchos de banda de red, identificar y resolver problema de recursos.

# Objetivos (II)

- **Áreas clave de conocimiento:** Medir uso de la CPU, Medir uso de la Memoria, Medir operaciones de entrada/salida en disco, Medir operaciones de entrada/salida en la red, Medir el rendimiento del cortafuegos y el enrutamiento, asignar el uso del ancho de banda del cliente, relacionar los síntomas del sistema con los problemas probables, estimar el rendimiento e identificar los cuellos de botella en un sistema, incluida la red.

# Objetivos (III)

- **Términos y utilidades:** iostat, netstat, w, top. sar, processes blocked on I/O, blocks out. Vmstat, pstree, ps,lsof,uptime,swap, blocks in

# Índice

1. Introducción
2. CPU
3. Memoria
4. Disco
5. Red
6. Otras herramientas
7. Resolución de Problemas



# Introducción

---

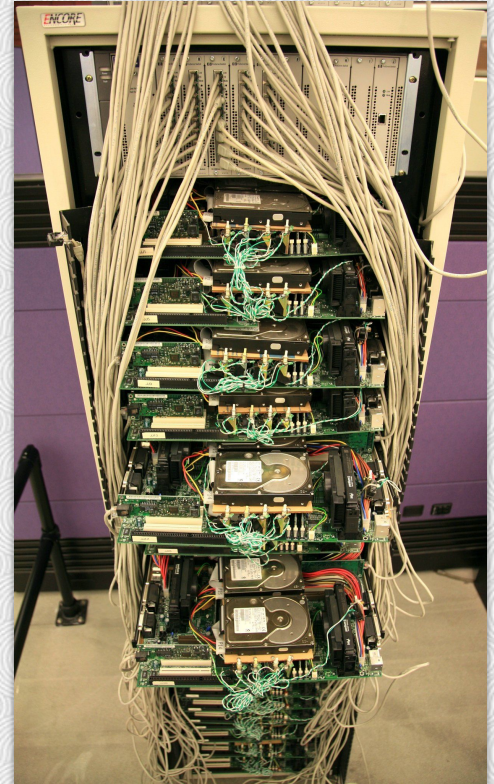
1

---

## ¿Qué es un administrador de sistemas (AS)?

- ★ Es la persona que tiene la responsabilidad de implementar, configurar, mantener, **monitorizar**, documentar y asegurar el correcto funcionamiento de un sistema informático, o algún aspecto de este.

Google's First Production Server





# ¿Qué es un administrador de sistemas (AS)? (II)



How IT people see each other

## ¿Por qué monitorizar?

- ★ Es una de las tareas más crítica, y frecuentes (proactivos), que debe realizar un AS
- ★ Permite detectar problemas de seguridad o de rendimiento, entre otros, dentro del sistema.

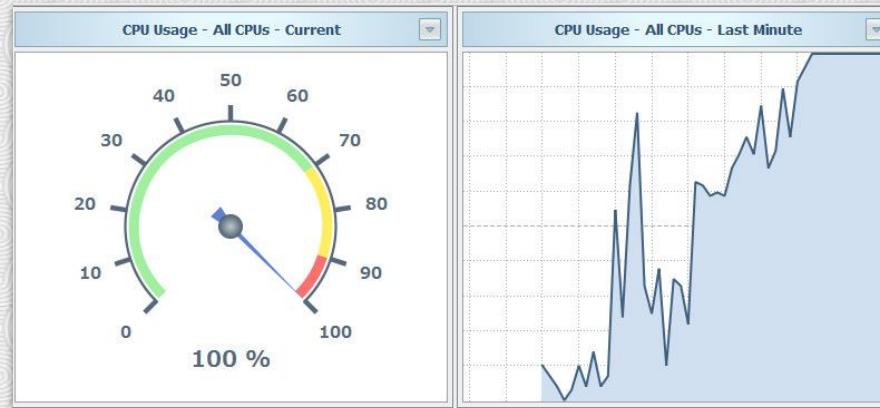


## ¿Qué y cómo monitorizamos?

- ★ Aplicamos una serie de medidas sobre un conjunto de recursos
- ★ Los recursos que monitorizamos son:
  - **CPU**
  - **Memoria**
  - **Disco**
  - **Red**
- ★ Buscamos recopilar datos sobre: Tiempo de actividad del sistema, estadísticas de uso y carga de la CPU, estadísticas de uso de la memoria y swap, estadísticas de carga y E/S del disco, estadísticas de carga y E/S de la red, etc.

# Objetivos

- ★ Monitorizar el sistema para detectar problemas de rendimiento
- ★ Monitorizar el rendimiento de un programa específico
- ★ Monitorizar el sistema para detectar problemas de seguridad





# CPU

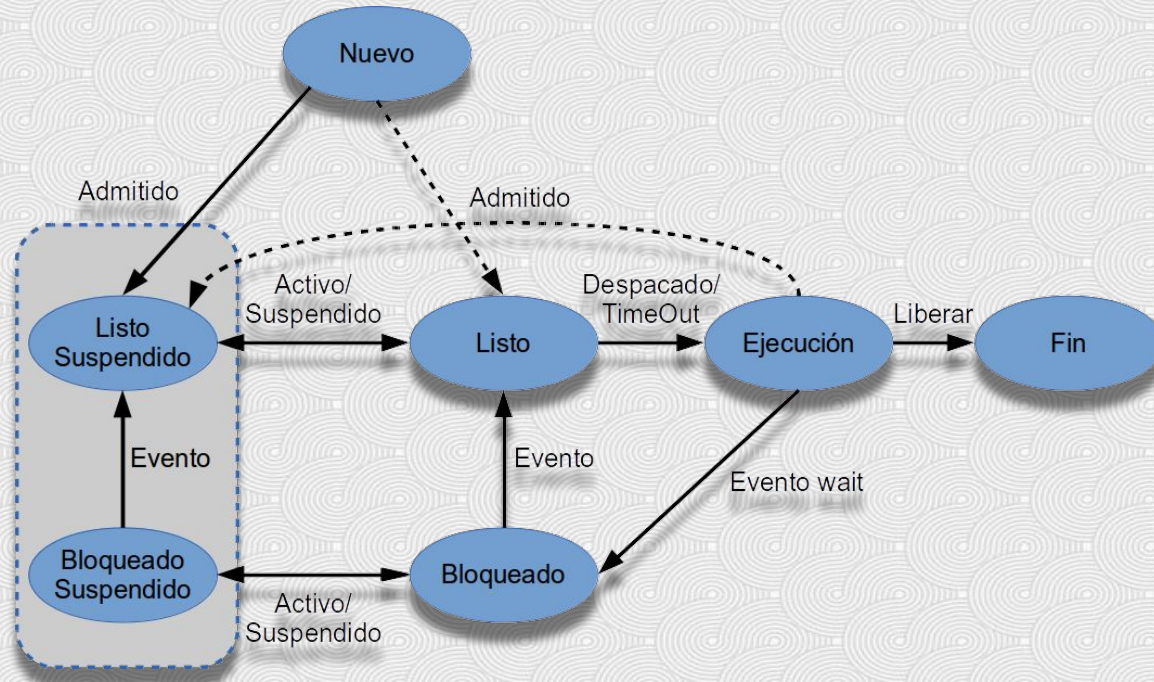


2



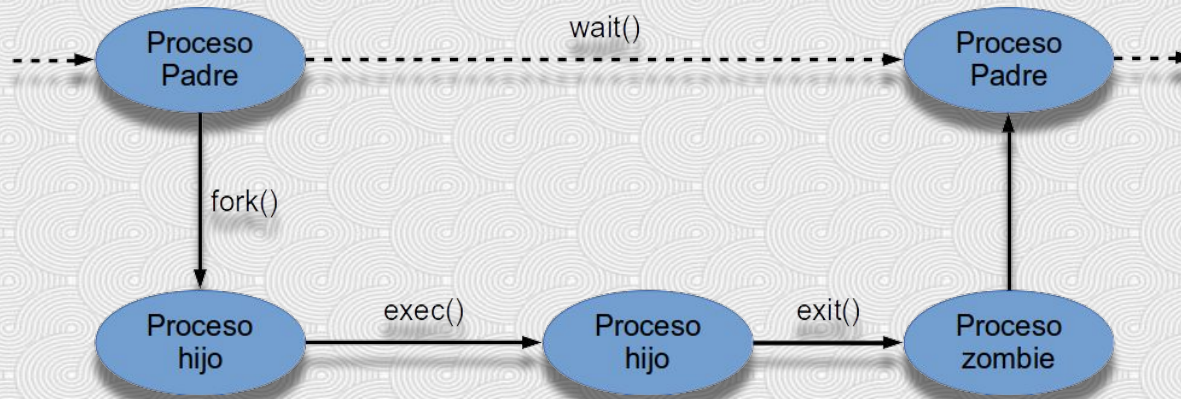
## ¿Qué es?

★ Es la encargada de ejecutar los programas (procesos)



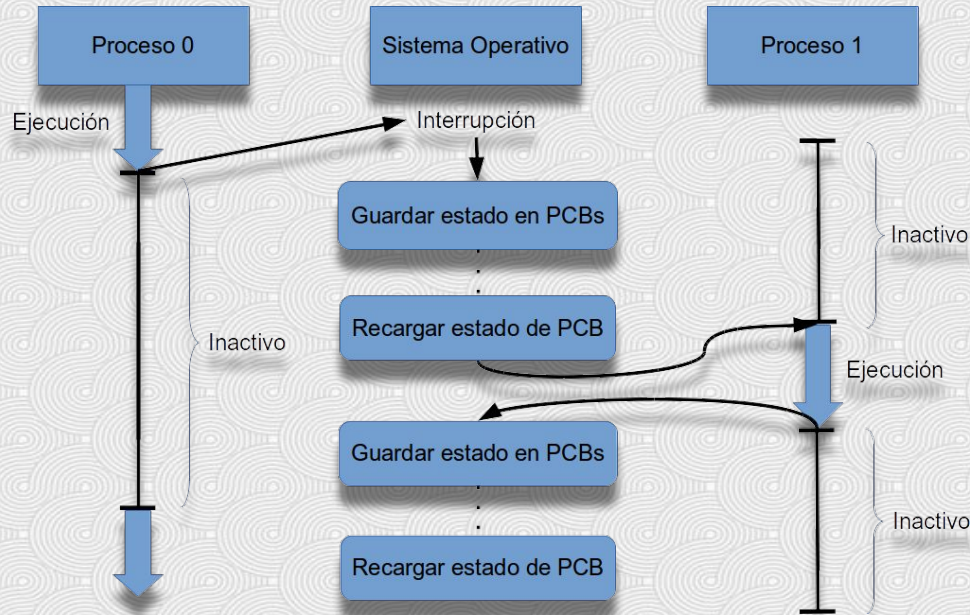
## El proceso zombie

- ★ Es un proceso que ha acabado su ejecución, pero aún tiene entrada en la tabla de procesos



# Cambios de contexto y cambios de proceso

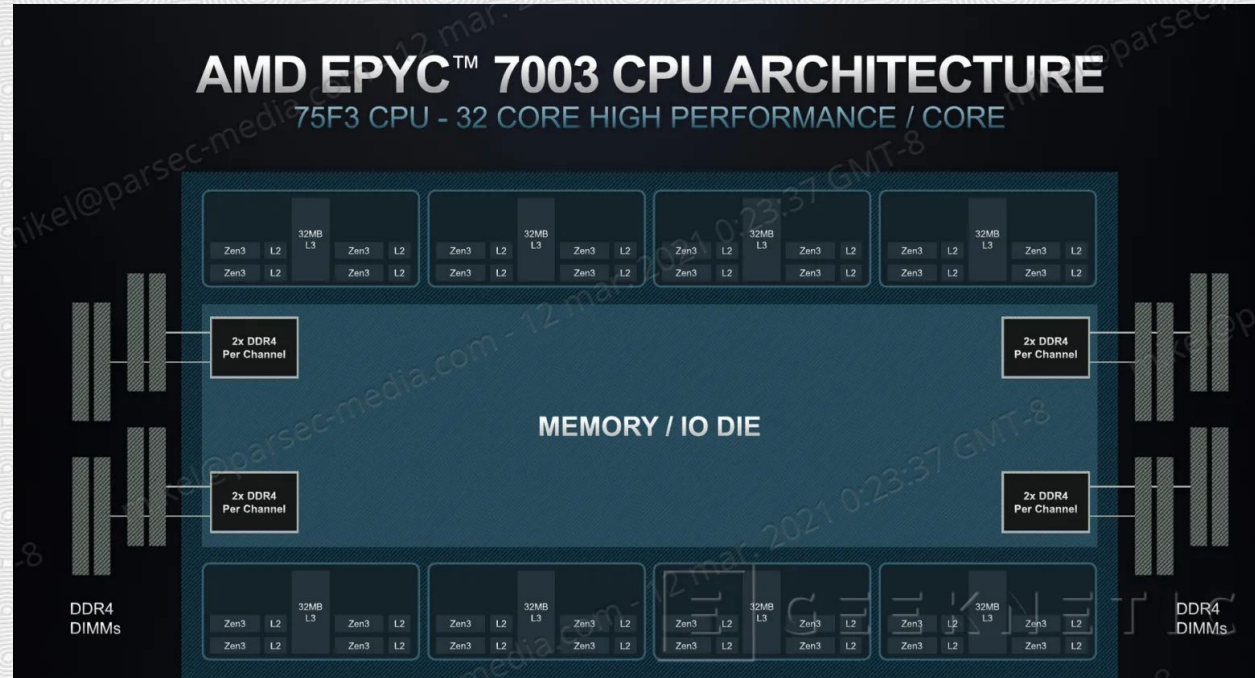
- ★ Cuando un proceso libera la CPU se produce un cambio de contexto y, en ocasiones, un cambio de proceso



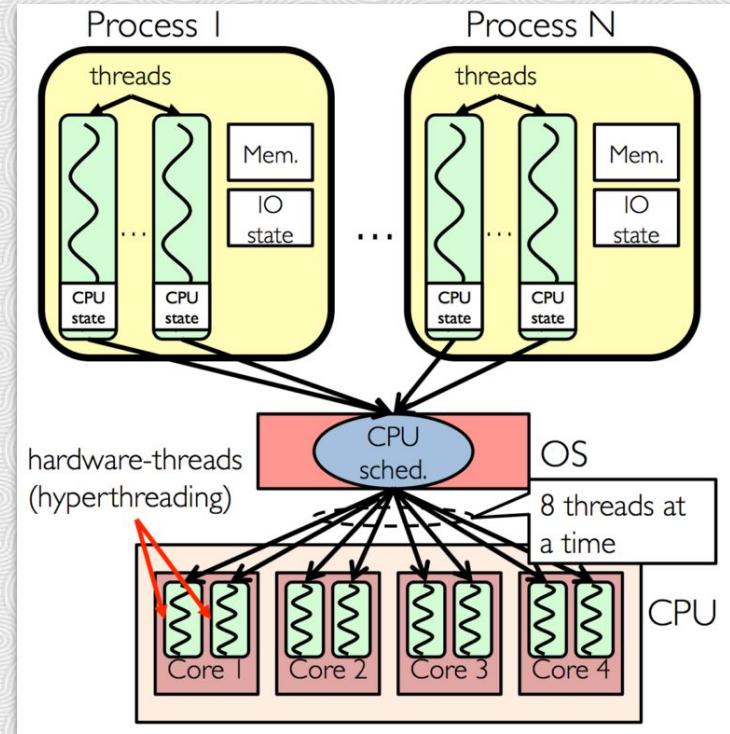
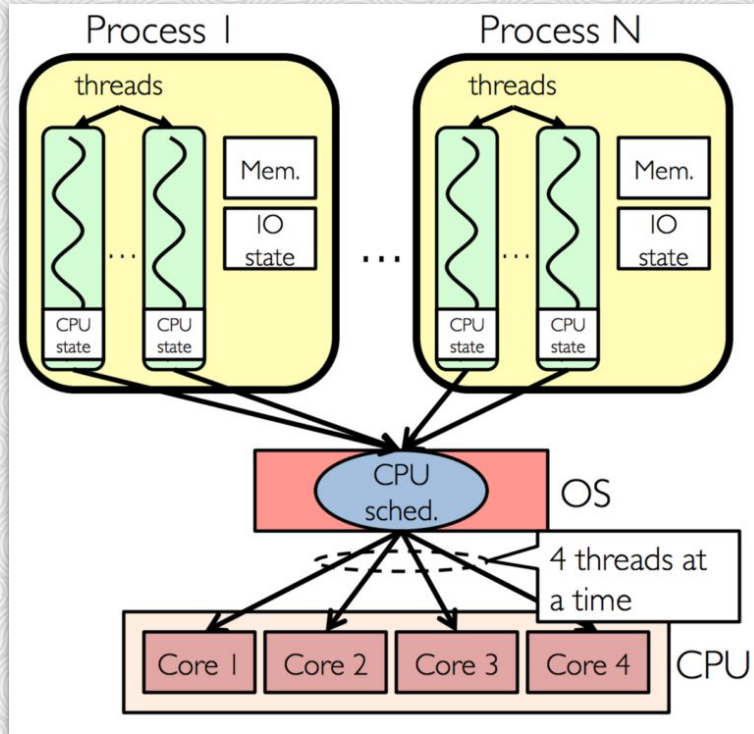


# Microprocesadores actuales: multicore, multihebra

- ★ AMD EPYC 7763
  - Cores: 64
  - Threads: 128
  - Precio: 1500 € (febrero 2021)
- ★ [www.cpubenchmarkmark.net](http://www.cpubenchmarkmark.net)
- ★ Un servidor tiene más de un slot (microprocesador)



# Multi Core vs. Hyper-threading (procesos vs hebras)





# Medidas

- ★ Además del número de procesos por cola, o el de cambios de contexto, la medida más usada son:
  - **La carga del sistema (load average).**
  - **El tiempo que la CPU está activa o inactiva**

## Carga del sistema

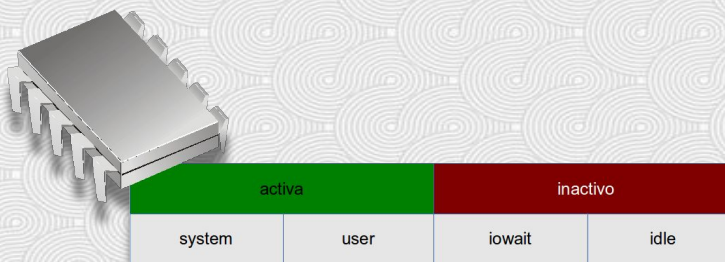
- ★ Representa la media ponderada del número de procesos en la cola de preparados en los últimos 1, 5 y 15 minutos. Se calcula cada 4 segundos

```
[root@localhost ~]# uptime  
11:29:27 up 1:54, 1 user, load average: 0,37, 0,57, 0,68
```

- ★ Un valor superior al 1 indicaría una sobrecarga de la CPU.
- ★ Si tenemos un sistema con 2 CPU, un valor de 1,5 no indicaría sobrecarga. Un valor de 2,5 sí lo indicaría.
- ★ Esto es, el valor de la carga media hay que dividirlo entre el total de CPU para determinar si el sistema está sobrecargado.

## Estado de la CPU

- ★ La CPU puede estar activa o inactiva. Cada uno de estos estados se divide en:



```
[root@localhost ~]# iostat -c
Linux 5.8.11-1-MANJARO (localhost)      05/10/20      _x86_64_      (8 CPU)
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5,32    0,01  1,83    2,86    0,00   89,98
```

# Herramientas

★ Las herramientas más usadas para medir la CPU son:

- uptime
- iostat
- vmstat
- ps
- pstree
- w
- top

## uptime

- ★ El comando `uptime` muestra el tiempo total que lleva el sistema ejecutándose así como la carga media del sistema:

```
[root@localhost ~]# uptime  
16:17:01 up 8:03, 2 users, load average: 0,59, 0,56, 0,53
```

Hora actual

Periodo sin reinicios

Usuarios conectados

Carga del sistema



## iostat

- ★ El comando **iostat** se usa para monitorizar dispositivos de E/S pero la opción **-c** nos permite mostrar aspectos específicos de la CPU:

```
[root@localhost ~]# iostat -c
Linux 5.8.11-1-MANJARO (localhost)      05/10/20      _x86_64_      (8 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5,32    0,01   1,83    2,86    0,00   89,98
```



...

## ¿Qué información devuelve? (I)

```
[root@localhost ~]# iostat -c
```

```
Linux 5.8.11-1-MANJARO (localhost)
```

```
05/10/20
```

```
_x86_64_
```

```
(8 CPU)
```

avg-cpu:	%user	%nice	%system	%iowait	%steal	%idle
	5,32	0,01	1,83	2,86	0,00	89,98

Tipo de núcleo, versión y nombre del host

Fecha actual

Arquitectura

Número de CPU

## ¿Qué información devuelve? (II)

```
[root@localhost ~]# iostat -c
Linux 5.8.11-1-MANJARO (localhost)      05/10/20    _x86_64_      (8 CPU)
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5,32    0,01   1,83    2,86    0,00   89,98
```

- **%user**: % de tiempo en el que la CPU está en modo usuario.
- **%nice**: % de tiempo dedicado al cambio de prioridad de procesos.
- **%system**: % de tiempo en el que la CPU está en modo usuario.
- **%iowait**: % de tiempo en el que la CPU está esperando operaciones E/S.
- **%steal**: % de tiempo esperando a que el hipervisor gestione solicitudes a la CPU virtual.
- **%idle**: % de tiempo en el que la CPU no gestiona ninguna solicitud

## Ejecución iterativa

- El comando **iostat** permite ejecutarse **cada** cierto intervalo de tiempo hasta un **número máximo** de intervalos:

```
[root@localhost ~]# iostat -c 1 3
Linux 5.8.12-989.native (localhost)      05/10/20      _x86_64_      (16 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0,32  0,00  0,11  0,03  0,00    99,54

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0,25  0,00  0,31  0,00  0,00    99,44

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0,06  0,00  0,19  0,19  0,00    99,56
```

## vmstat

- ★ El comando **vmstat** devuelve información, principalmente, de la memoria virtual aunque también aporta información sobre el estado de la CPU

```
[root@localhost ~]# vmstat
```

procs		-----memory-----				---swap--		-----io----		-system-		-----cpu-----				
r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
0	0	0	27415816	461608	2080264	0	0	18	7	88	167	0	0	99	0	0



## ¿Qué información devuelve? (I)

- ★ Las columnas bajo la cabecera procs hacen referencia a procesos
  - **r**: número de procesos ejecutándose o esperando a ejecutarse
  - **b**: número de procesos en sueño ininterrumpible (esperando a operación de E/S)

```
[root@localhost ~]# vmstat
```

```
procs -----memory----- ---swap-- -----io----- -system-- -----cpu-----  
r  b    swpd    free    buff    cache    si    so    bi    bo    in    cs    us    sy    id    wa    st  
0  0         0 27415816 461608 2080264    0    0    18    7    88   167    0    0 99    0    0
```

## ¿Qué información devuelve?(II)

- ★ Las columnas bajo la cabecera **system** hacen referencia a los siguientes aspectos del sistema
  - **in**: número de interrupciones por segundo
  - **cs**: cambios de contexto por segundo

```
[root@localhost ~]# vmstat
```

```
procs -----memory----- --swap--  -----io----- -system-  -----cpu-----  
r  b   swpd   free   buff  cache   si   so   bi   bo   in   cs  us  sy  id  wa  st  
0  0     0 27415816 461608 2080264    0    0   18   7   88  167  0  0 99  0  0
```

## ¿Qué información devuelve?(II)

- ★ Las columnas bajo la cabecera **cpu** muestras aspectos relacionados con la CPU (en %):
  - ★ **us**: tiempo dedicado a tareas de usuario
  - ★ **sy**: tiempo dedicado a tareas del sistema
  - ★ **id**: tiempo de inactividad
  - ★ **wa**: tiempo cpu bloqueada
  - ★ **st**: tiempo que el hypervisor dedica a gestionar peticiones virtuales

```
[root@localhost ~]# vmstat
```

```
procs -----memory----- --swap-- -----io----- -system- -----cpu-----  
r  b   swpd   free   buff  cache   si   so    bi   bo    in   cs   us  sy  id  wa  st  
0  0     0 27415816 461608 2080264    0    0    18   7   88  167  0  0 99  0  0
```

## Ejecución iterativa

- ★ El comando **vmstat** permite ejecutarse **cada** cierto intervalo de tiempo hasta un **número máximo** de intervalos:

```
[root@localhost ~]# vmstat 5 4
```

```
procs -----memory----- ---swap-- ----io---- -system-- -----cpu-----
 r  b    swpd   free   buff  cache   si   so    bi   bo    in   cs us  sy id wa st
 1  0      0 23496316 585080 3813844    0    0   24   7  108  207  0  0 99  0  0
 3  0      0 23069052 585156 3816356    0    0    0  789 10060 29571  7  2 91  0  0
 0  0      0 22528040 585160 3817512    0    0    6   48 25106 35148 27  3 70  0  0
 0  0      0 23502232 585172 3819548    0    0  558 11567 3245 6236  1  0 98  1  0
```

## ps

- ★ El comando **ps** muestra los procesos que se están ejecutando actualmente en el sistema. La opción **-e** muestra todos los procesos:

```
[root@localhost ~]# ps -e
  PID TTY          TIME CMD
    1 ?            00:00:00 systemd
    2 ?            00:00:00 kthreadd
    3 ?            00:00:00 rcu_gp
 5453 pts/0    00:00:00 bash
 1874 tty2      00:00:00 gnome-session-b
```

- ★ La opción **-a** muestra los procesos con TTY asociada.



## Relaciones entre padre/hijo (I)

- ★ Si queremos mostrar información sobre relaciones padre/hijo e incluir el tiempo de uso de la CPU usamos **-H**:

```
[root@localhost ~]# ps -eHo pid, ppid cmd
PID  PPID  CMD
557  1     /usr/sbin/gdm
1823 557    gdm-session-worker [pam/gdm-password]
1847 1823   /usr/libexec/gdm-x-session --run-script /usr/bin/gnome-session
1849 1847   /usr/bin/X vt2 -displayfd 3 -auth /run/user/1000/gdm/Xauthority
-nolisten tcp -background none -noreset -keeptty -novtswitch -verbose 3
```

- ★ La opción **-o** permite indicar qué información queremos ver del proceso. Es muy habitual la combinación **-efH**.

## Relaciones entre padre/hijo (II)

- ★ Una alternativa para ver las relaciones padre/hijo es usar el comando **pstree**, muestra estas relaciones en forma de árbol:

```
[root@localhost ~]# pstree 557
gdm├──gdm-session-wor├──gdm-x-session├──X──20*[{X}]
│                   │               │
│                   │               ├──gnome-session-b├──ssh-agent
│                   │               │               │
│                   │               │               └──3*[{gnome-session-b}]
│                   │               └──2*[{gdm-x-session}]
│                   └──2*[{gdm-session-wor}]
└──2*[{gdm}]
```

- ★ **N\*[algo]**: indica N procesos llamados algo
- ★ **N\*[{algo}]**: indica N hebras ejecutando algo

## W

★ El comando **w** muestra quién está conectado y qué está ejecutando.

```
[root@localhost ~]# w
12:14:15 up 33 days, 15:09,  2 users,  load average: 0.06, 0.12, 0.09
USER      TTY      FROM          LOGIN@  IDLE   JCPU   PCPU WHAT
root      tty2      -             17Jan08 18days 0.29s   0.04s login -- root
root      pts/0    bob           Sat22   0.00s   0.57s   0.56s -bash
```

## ¿Qué información devuelve? (I)

- ★ La primera línea devuelve lo mismo que el comando uptime.

```
[root@localhost ~]# w
12:14:15 up 33 days, 15:09,  2 users,  load average: 0.06, 0.12, 0.09
USER      TTY      FROM          LOGIN@  IDLE   JCPU   PCPU WHAT
root      tty2      -             17Jan08 18days 0.29s   0.04s login -- root
root      pts/0    bob           Sat22   0.00s   0.57s   0.56s -bash
```

## ¿Qué información devuelve? (II)

- ★ **USER y TTY**: nombre usuario y terminal de conexión
- ★ **FROM y LOGIN**: localización y cuándo se conectó
- ★ **IDLE**: tiempo inactivo
- ★ **JCPU y PCU**: tiempo de CPU usado y tiempo de CPU usado por el proceso actual
- ★ **WHAT**: proceso actual.

```
[root@localhost ~]# w
12:14:15 up 33 days, 15:09,  2 users,  load average: 0.06, 0.12, 0.09
USER      TTY      FROM          LOGIN@      IDLE        JCPU        PCPU WHAT
root      tty2      -             17Jan08     18days     0.29s       0.04s login -- root
root      pts/0     bob          Sat22       0.00s       0.57s       0.56s -bash
```



## top

- ★ El comando **top** proporciona una visión en tiempo real de un sistema en ejecución.
- ★ Esta utilidad muestra la misma información de los comandos anteriores. Además, permite: ordenar, filtran, envío de señales, etc.

# Memoria

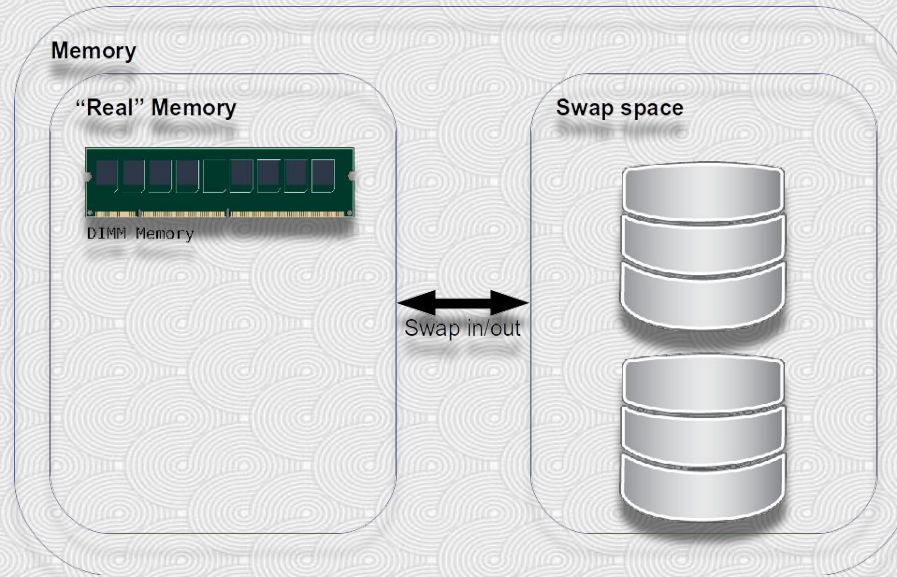
---

3

---

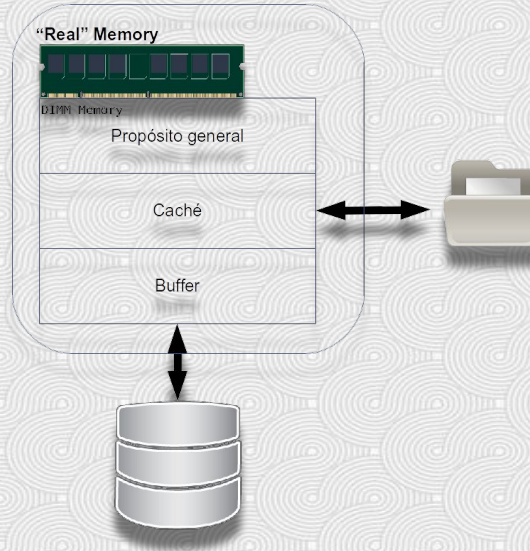
## ¿Qué es?

- ★ La memoria es el espacio en el que se albergan los procesos.
- ★ Esta se divide en memoria RAM y de intercambio



## ¿Qué es?

- ★ La memoria RAM se divide en de **propósito general, caché y buffer**



- ★ El tamaño de la caché y el buffer es dinámico

# Medidas

- ★ Se suele medir el % de espacio libre y ocupado en:
  - Memoria principal
    - Propósito general
    - Caché
    - Buffers
  - Memoria de intercambio



# Herramientas

- ★ Las herramientas más usadas para medir la memoria son:
  - free
  - vmstat

## free

- ★ El comando **free** muestra información sobre la memoria física, la memoria de swap, así como los buffers y caché usados por el sistema:

```
[root@localhost ~]# free
```

	total	used	free	shared	buff/cache	available
Mem:	32818776	5628288	22084828	937664	5105660	25794012
Swap:	65532	0	65532			

- ★ La opción **-k**, **-m**, **-g** nos permite cambiar las unidades. **-h** ajusta las unidades automáticamente

```
[root@localhost ~]# free -m
```

	total	used	free	shared	buff/cache	available
Mem:	32049	5617	21410	916	5021	25067
Swap:	63	0	63			

## ¿Qué información devuelve? (I)

- ★ **total**: Total de memoria (RAM y Swap)
- ★ **used**: Memoria usada (calculada como total - free - buffers - cache)
- ★ **free**: Memoria sin usar (RAM y Swap)

```
[root@localhost ~]# free -m
```

	total	used	free	shared	buff/cache	available
Mem:	32049	5617	21410	916	5021	25067
Swap:	63	0	63			

## ¿Qué información devuelve? (II)

- ★ **shared**: Memoria usada por **tmpfs**
- ★ **buffers**: Memoria usada por los buffers del kernel
- ★ **cache**: Memoria usada por la caché
- ★ **buff/cache**: Suma de buffers y caché
- ★ **available**: Estimación de la cantidad de memoria disponible para ejecutar nuevas aplicaciones

```
[root@localhost ~]# free -m
```

	total	used	free	shared	buff/cache	available
Mem:	32049	5617	21410	916	5021	25067
Swap:	63	0	63			

## vmstat

- ★ El comando **vmstat** muestra información sobre la memoria física y de swapping:

```
[root@localhost ~]# vmstat
-----memory----- ---swap--
swpd  libre  búfer  caché    si   so
0     9840808 1243776 2172964    0    0
```

- ★ La opción **-s**, permite obtener unas estadísticas más detalladas de la memoria, la cpu, etc.



## ¿Qué información devuelve?

- ★ **swpd**: cantidad de memoria virtual usada.
- ★ **free**: cantidad de memoria libre.
- ★ **buff**: cantidad de memoria usada para buffers.
- ★ **cache**: cantidad de memoria usada para cache.
- ★ **si**: Cantidad de memoria intercambia desde disco.
- ★ **so**: Cantidad de memoria intercambia al disco.

```
[root@localhost ~]# vmstat
-----memory-----  ---swap--
swpd  libre   búfer   caché      si    so
0     9840808 1243776 2172964     0     0
```

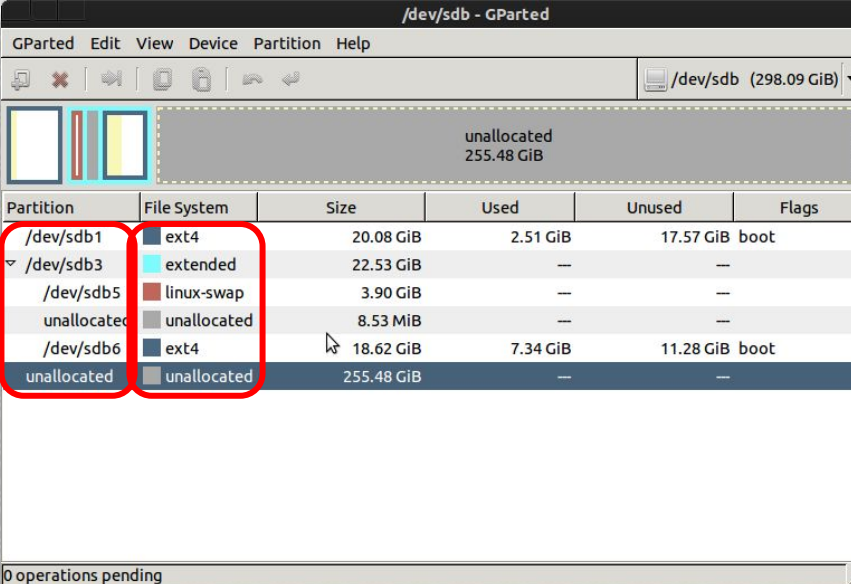
# Disco



4

## ¿Qué es?

- ★ Dispositivos de almacenamiento permanente
- ★ El SO hace uso de estos dispositivos mediante sistemas de ficheros.



GParted /dev/sdb - GParted

GPtred Edit View Device Partition Help

/dev/sdb (298.09 GiB)

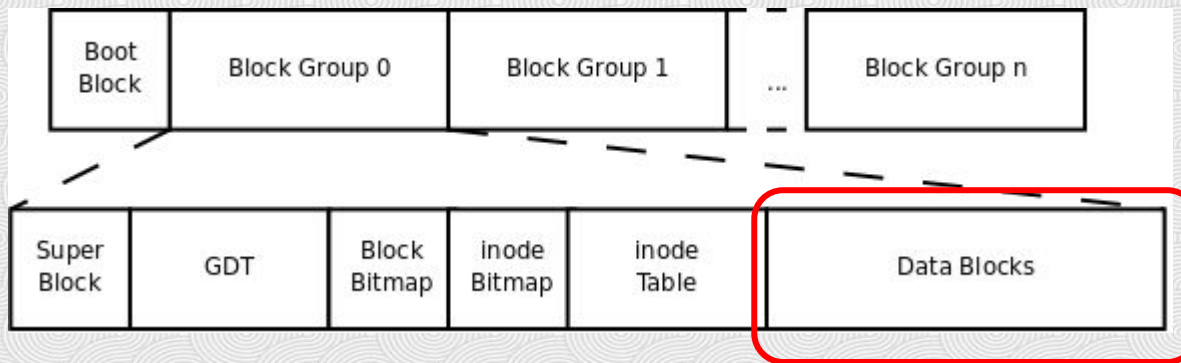
unallocated 255.48 GiB

Partition	File System	Size	Used	Unused	Flags
/dev/sdb1	ext4	20.08 GiB	2.51 GiB	17.57 GiB	boot
▼ /dev/sdb3	extended	22.53 GiB	—	—	
/dev/sdb5	linux-swap	3.90 GiB	—	—	
unallocated	unallocated	8.53 MiB	—	—	
/dev/sdb6	ext4	18.62 GiB	7.34 GiB	11.28 GiB	boot
unallocated	unallocated	255.48 GiB	—	—	

0 operations pending

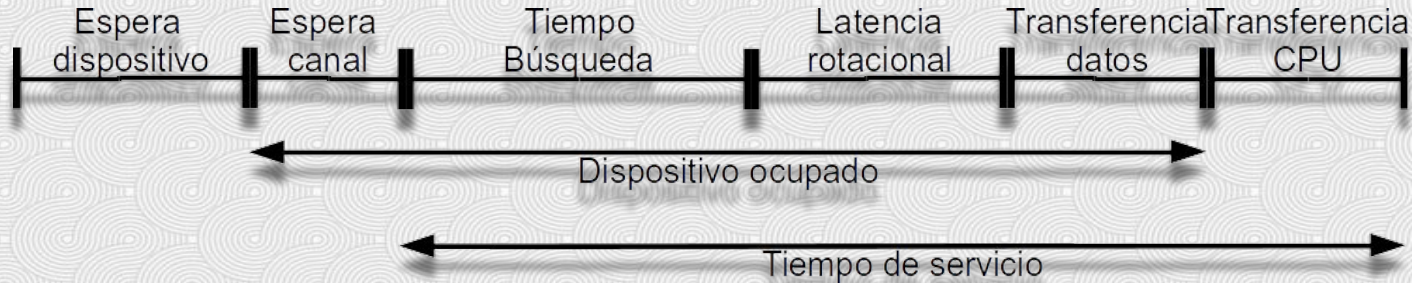
# Medidas

- ★ Se suele medir:
  - El % de espacio usado/ocupado.
  - La velocidad de acceso
- ★ No todo el espacio está disponible para el usuario, parte del espacio es usado por el sistema para tareas internas



# Medidas

- ★ **Tiempo de espera:** El núcleo pone la petición en la cola y espera a obtener el resultado



- ★ El núcleo reordena las peticiones de la cola para minimizar el tiempo de búsqueda



# Herramientas

★ Las herramientas más usadas para medir el disco son:

- iostat
- lsof
- vmstat
- df
- lsblk

## iostat

- ★ El comando **iostat** se usa para monitorizar los dispositivos de E/S (opción **-d** por defecto) monitorizando el tiempo que los dispositivos están activos en relación con su velocidad de transferencia media.

```
[root@localhost ~]# iostat
Linux 5.8.12-989.native (localhost)      05/10/20      _x86_64_      (16 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0,47  0,00  0,18  0,03  0,00   99,33

Device            tps    kB_read/s kB_wrtn/s  kB_dscd/s   kB_read  kB_wrtn  kB_dscd
nvme0n1           9,25       150,64    97,86      2,77     3560697  2313144  65532
sda                0,02         0,25      0,00      0,00        5797      0         0
sdb                0,02         0,20      0,00      0,00        4762     24         0
```

## ¿Qué información devuelve? (I)

- ★ **device**: nombre del dispositivo
- ★ **tps**: número de transferencias por segundo
- ★ **Blk\_read/s** y **Blk\_wrtn/s**: bloques leídos/escritos por segundo
- ★ **B\_dscd/s**: bloques descartados por segundo
- ★ **Blk\_read** y **Blk\_wrtn** bloques totales leídos/escritos
- ★ **kB\_dscd**: Bloque totales descartados.

```
[root@localhost ~]# iostat
```

```
...
```

Device	tps	kB_read/s	kB_wrtn/s	kB_dscd/s	kB_read	kB_wrtn	kB_dscd
nvme0n1	9,25	150,64	97,86	2,77	3560697	2313144	65532
sda	0,02	0,25	0,00	0,00	5797	0	0
sdb	0,02	0,20	0,00	0,00	4762	24	0

## ¿Qué información devuelve? (II)

- ★ Las medidas son en bloque de 512 bytes, podemos cambiar las unidades usando las opciones **-k**, **-m** o **-h**.

```
[root@localhost ~]# iostat -h
```

tps	kB_read/s	kB_wrtn/s	kB_dscd/s	kB_read	kB_wrtn	kB_dscd	Device
8,92	144,6k	95,8k	2,7k	3,4G	2,3G	64,0M	nvme0n1
0,01	0,2k	0,0k	0,0k	5,7M	0,0k	0,0k	sda
0,01	0,2k	0,0k	0,0k	4,7M	24,0k	0,0k	sdb

- ★ La opción **-p** muestra la misma información desglosada por particiones
- ★ Si usamos la opción **-x**, obtendremos más información (sectores leídos/escritos por segundo, solicitudes mezcladas por segundo...)

# lsof

- ★ El comando **lsof** muestra los ficheros abiertos por los procesos
- ★ Cuando hablamos de ficheros nos referimos a: un fichero regular, un directorio, un fichero especial de bloques, un fichero especial de caracteres, una librería, un flujo o fichero de red...

```
[root@localhost ~]# lsof
COMMAND  PID    TID TASKCMD  USER   FD      TYPE      DEVICE  SIZE/OFF      NODE NAME
systemd   1      systemd root    cwd     DIR      259,2    4096          2 /
systemd   1      systemd root    rtd     DIR      259,2    4096    16447671
/usr/lib64/libffi.so.6.0.4
```



## ¿Qué información devuelve? (I)

- ★ **COMMAND** y **PID**: el comando que abre el fichero y su identificador de proceso
- ★ **TID** y **TASKCMD**: identificador de la hebra y la tarea que ejecuta
- ★ **USER**: usuario que ejecuta el proceso

```
[root@localhost ~]# lsof
COMMAND      PID      TID  TASKCMD  USER
systemd      1
systemd      1
Gnome-key    1729     7229  dispatch root
```

# Comandos útiles



```
[root@localhost ~]# lsof /var/log/syslog # procesos que tienen abierto un fichero
[root@localhost ~]# lsof +d /var/log/ # ficheros abiertos en un directorio
[root@localhost ~]# lsof -c ssh -c init # ficheros abiertos por procesos que empiezan
por la cadena
[root@localhost ~]# lsof /home # procesos que están usando un punto de montaje
[root@localhost ~]# lsof -u lakshmanan # ficheros abiertos por el usuario
[root@localhost ~]# lsof -u ^lakshmanan # ficheros abiertos por otro usuario al
indicado
[root@localhost ~]# lsof -p 1753 # ficheros abiertos por un proceso
[root@localhost ~]# lsof -i # solo procesos que tenga abiertas conexiones de red
[root@localhost ~]# lsof -i 4# solo procesos que tenga abiertas conexiones de red TCP
[root@localhost ~]# lsof -i TCP:22# solo procesos que tenga abiertas conexiones de red
TCP en el puerto 22
```

## vmstat

- El comando **vmstat** también puede dar información general de E/S

```
[root@localhost ~]# vmstat
```

procs		-----memory-----				---swap--		-----io----		-system--		-----cpu-----				
r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	st
0	0	131584	320832	491836	3944672	0	0	10	14	33	58	3	1	96	0	0

- ★ Si usamos la opción **-d**, podemos obtener información detallada de las operaciones de E/S por dispositivos (**-p** para particiones):

```
[root@localhost ~]# vmstat -d
```

disk-	-----reads-----				-----writes-----				-----IO-----	
	total	merged	sectors	ms	total	merged	sectors	ms	cur	sec
sda	371911	42759	7909491	2062907	288793	312514	13208648	2693192	0	2652

## ¿Qué información devuelve? (I)

- **bi**: bloques por segundo leídos en todos los dispositivos
- **bo**: bloques por segundo escritos en todos los dispositivos

```
[root@localhost ~]# vmstat
procs  -----memory-----  ---swap--  -----io-----  -system--  -----cpu-----
 r   b   swpd   free   buff   cache   si    so    bi    bo    in    cs  us  sy  id  wa  st
  0   0 131584 320832 491836 3944672    0    0    10    14   33   58   3   1  96   0   0
```

## ¿Qué información devuelve? (II)

- **disk**: nombre del dispositivo
- **total**: total de lecturas/escrituras realizadas
- **merged**: lecturas/escrituras agrupadas (una operación de E/S)
- **sectors**: sectores leídos/escritos
- **ms**: milisegundos consumidos en leer/escribir.
- **cur**: operación de E/S en progreso
- **sec**: segundos consumidos en cada operación de E/S.

```
[root@localhost ~]# vmstat -d
```

```
disk- -----reads----- -----writes----- ----IO-----  
      total      merged    sectors    ms        total merged sectors    ms    cur   sec  
sda  371911    42759    7909491  2062907  288793 312514 13208648 2693192    0   2652
```



# df

- ★ El comando **df** muestra la cantidad de espacio disponible en los distintos sistemas de ficheros:

```
[root@localhost ~]# df
S.ficheros    Tamaño Usados  Disp Uso% Montado en
dev           7,8G      0  7,8G   0% /dev
run           7,8G    2,0M  7,8G   1% /run
/dev/sdb2     110G  95G   9,2G  92% /
tmpfs         7,8G    139M  7,7G   2% /tmp
/dev/sdb1     300M    280K  300M   1% /boot/efi
/dev/loop0    218M    218M   0 100% /var/lib/snapd/snap/gnome-3-34-1804/60
/dev/sda7     1,6T    1,3T  218G  86% /home
```

## ¿Qué información devuelve?

- ★ **S.ficheros**: fichero de dispositivo
- ★ **Tamaño**: Tamaño total
- ★ **Usados, Disp, Uso%**: cantidad usada, disponible y % de uso
- ★ **Montado en**: punto de montaje

```
[root@localhost ~]# df
```

S.ficheros	Tamaño	Usados	Disp	Uso%	Montado en
dev	7,8G	0	7,8G	0%	/dev
run	7,8G	2,0M	7,8G	1%	/run
/dev/sdb2	110G	95G	9,2G	92%	/
tmpfs	7,8G	139M	7,7G	2%	/tmp
/dev/sdb1	300M	280K	300M	1%	/boot/efi
/dev/loop0	218M	218M	0	100%	/var/lib/snapd/snap/gnome-3-34-1804/60
/dev/sda7	1,6T	1,3T	218G	86%	/home

# lsblk

- ★ El comando **lsblk** muestra información sobre dispositivos de bloques
- ★ La información la obtiene del **sysfs** y **udev db**.

```
[root@localhost ~]# lsblk
```

```
lsblk
```

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINT
loop0	7:0	0	217,9M	1	loop	/var/lib/snapd/snap/gnome-3-34-1804/60
sda	8:0	0	1,8T	0	disk	
└─sda1	8:1	0	1023M	0	part	
└─sda2	8:7	0	1,5T	0	part	/home
└─sda3	8:8	0	15,9G	0	part	[SWAP]ijfviana/Driver

## ¿Qué información devuelve?

- ★ **NAME**: nombre del dispositivo
- ★ **MAJ:MIN**: número mayor y menor
- ★ **SIZE**: Tamaño total
- ★ **RO**: sólo lectura
- ★ **TYPE** y **MOUNTPOINT**: Tipo de dispositivo y punto de montaje

```
[root@localhost ~]# lsblk
```

```
lsblk
```

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINT
loop0	7:0	0	217,9M	1	loop	/var/lib/napd/snap/gnome-3-34-1804/60
sda	8:0	0	1,8T	0	disk	
└─sda1	8:1	0	1023M	0	part	
└─sda2	8:7	0	1,5T	0	part	/home
└─sda3	8:8	0	15,9G	0	part	[SWAP]

# Redes

---

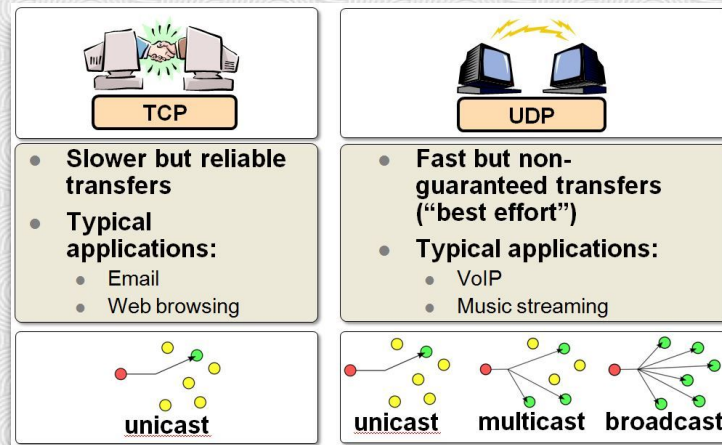
5

---



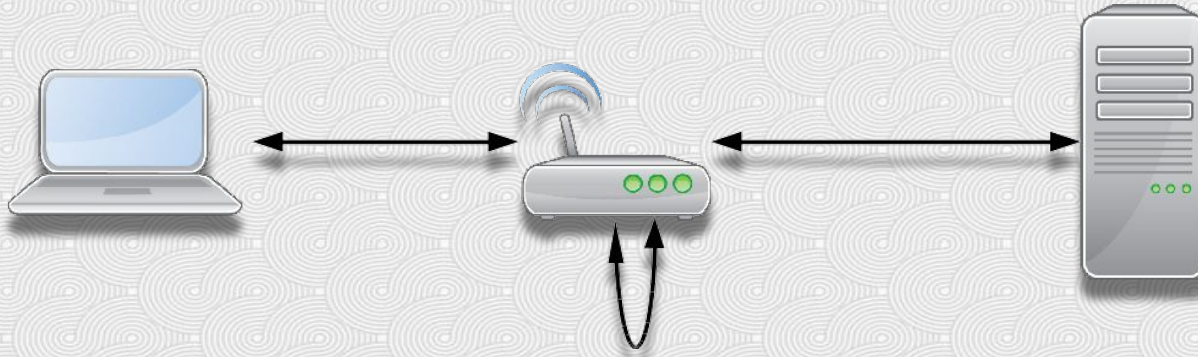
## ¿Qué es?

- ★ Es la parte del sistema operativo que se encarga de gestionar la comunicación entre procesos
- ★ Estos procesos puede residir en la misma máquina (unix sockets) o en máquina distintas (tcp, udp sockets)



# Medidas

- ★ Medimos los paquetes en la comunicación
  - cliente-servidor
  - enrutado
- ★ Las medidas que se usan son bit por segundo o paquetes por segundo (menos fiable)



# Herramientas

- ★ Las herramientas más usadas para medir la memoria son:
  - netstat

## netstat (I)

- ★ El comando **netstat** (es desuso a favor de **ss** e **ip**) muestra información sobre conexiones (**-t** para tcp **-u** para udp):

```
[root@localhost ~]# netstat
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 localhost:cap           localhost:42604         TIME_WAIT
```

- ★ Tablas de enrutamiento con la opción **-r**:

```
[root@localhost ~]# netstat -r
Destination    Gateway         Genmask         Flags         MSS Window  irtt Iface
default        OpenWrt.lan    0.0.0.0         UG            0 0       0 eno1
172.17.0.0     0.0.0.0        255.255.0.0    U             0 0       0 docker0
192.168.1.0    0.0.0.0        255.255.255.0  U             0 0       0 eno1
```

## netstat (II)

★ 0 resúmenes estadísticos por protocolo, opción **-s**:

```
[root@localhost ~]# netstat -s
Ip:
    Forwarding: 1
    2365311 total packets received
    14 with invalid addresses
    0 forwarded
    0 incoming packets discarded
    2347606 incoming packets delivered
    1815815 requests sent out
    40 dropped because of missing route
```



## ¿Qué información devuelve? (I)

- ★ **Proto**: indica el protocolo usado en la conexión
- ★ **Recv-Q** y **Send-Q** paquetes recibidos /enviados.
- ★ **Local Address** y **Foreign Address**: nuestro extremo y extremo remoto d la conexión (incluido puerto)
- ★ **State**: estado de la conexión (LISTEN, TIME\_WAIT,...)

```
[root@localhost ~]# netstat -tua
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp      0      0 localhost:cap           localhost:42604         TIME_WAIT
tcp      0      0 *:printer               *:.*                    LISTEN
tcp6     0      0 linda.lan:32799         23.97.215.12:https     ESTABLISHED
tcp      1      0 dsl-78-199-139.s:1199  www.xodiox.com:http    CLOSE_WAIT
udp      0      0 linda.lan:bootpc       OpenWrt.lan:bootpc     ESTABLISHED
udp6     0      0 linda.lan:43488        [::]:.*
```

## ¿Qué información devuelve? (II)

- ★ Si indicamos la opción **-p**, obtendremos el PID del proceso que está usando el socket:

```
[root@localhost ~]# netstat -tap
Active Internet connections (servers and established)
Local Address Foreign Address  State    PID/Program name
*:printer    *:.*          LISTEN   988/inetd
bigcat:8000   *:.*          LISTEN   1064/junkbuster
*:time       *:.*          LISTEN   988/inetd
*:x11        *:.*          LISTEN   1462/X
*:http       *:.*          LISTEN   1078/httpd
bigcat:domain *:.*          LISTEN   956/named
bigcat:domain *:.*          LISTEN   956/named
*:ssh        *:.*          LISTEN   972/sshd
```

# Comando útiles

```
[root@localhost ~]# netstat -a | more # To show both listening and
non-listening sockets.
[root@localhost ~]# netstat -at      # To list all tcp ports.
[root@localhost ~]# netstat -au      # To list all udp ports.
[root@localhost ~]# netstat -l       # To list only the listening ports.
[root@localhost ~]# netstat -lt      # To list only the listening tcp ports.
[root@localhost ~]# netstat -lu      # To list only the listening udp ports.
[root@localhost ~]# netstat -lx      # To list only the listening UNIX ports.
[root@localhost ~]# netstat -s       # To list the statistics for all ports.
[root@localhost ~]# netstat -st(TCP) # To list the statistics for TCP ports.
[root@localhost ~]# netstat -pt      # To display the PID and program names
[root@localhost ~]# netstat -c       # To print the netstat information continuously.
```

# Otras herramientas

---

6

---

## ¿Qué información devuelve? (II)

- top
- Htop
- Sar
- Mpstat
- Lotop



# top

- ★ El comando **top** permite a los usuarios monitorizar procesos y recursos del sistema.
- ★ La última versión de este comando es la 3.7 y se liberó en el año 2008

```
top - 04:33:30 up 17:16, 1 user, load average: 0.05, 0.01, 0.03
Tasks: 75 total, 2 running, 73 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.3%sy, 0.0%ni, 88.6%id, 0.0%wa, 0.0%hi, 11.1%si, 0.0%st
Mem: 515348k total, 319956k used, 195392k free, 43432k buffers
Swap: 1048568k total, 0k used, 1048568k free, 201748k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4	root	RT	-5	0	0	0	S	0.3	0.0	0:12.90	watchdog/0
32348	root	15	0	2200	996	796	R	0.3	0.2	0:00.04	top
1	root	15	0	2068	612	528	S	0.0	0.1	0:34.24	init
2	root	RT	-5	0	0	0	S	0.0	0.0	0:00.00	migration/0
3	root	34	19	0	0	0	S	0.0	0.0	0:00.04	ksoftirqd/0
5	root	10	-5	0	0	0	S	0.0	0.0	0:04.77	events/0
6	root	10	-5	0	0	0	S	0.0	0.0	0:01.84	khelper
7	root	11	-5	0	0	0	S	0.0	0.0	0:00.06	kthread
10	root	10	-5	0	0	0	S	0.0	0.0	0:00.72	kblockd/0
11	root	20	-5	0	0	0	S	0.0	0.0	0:00.00	kacpid
47	root	20	-5	0	0	0	S	0.0	0.0	0:00.00	cqueue/0
50	root	10	-5	0	0	0	S	0.0	0.0	0:00.00	khubd
52	root	10	-5	0	0	0	S	0.0	0.0	0:00.02	kseriod

# Htop

- ★ **htop** es un sistema de monitorización, administración y visor de procesos interactivo.

```

1  [||] 0.7% Tasks: 102, 165 thr; 1 running
2  [||] 0.5% Load average: 0.00 0.03 0.05
3  [||] 0.8% Uptime: 00:53:03
4  [||] 0.5%
Mem[|||||||||||||||||877/993MB]
Swp[|] 2/1534MB

  PID USER   PRI  NI  VIRT   RES   SHR  S  CPU% MEM%   TIME+  Command
3106 user    20   0  237M  3592  1984  S   0.0  0.4   0:00.08 xfce4-session
3200 user    20   0  463M 17640 1452  S   0.0  1.7   0:00.25 /usr/bin/python /
3267 user    20   0  153M  3940  2308  S   0.0  0.4   0:00.04 /usr/lib/x86_64-l
1501 root     20   0  211M   776   444  S   0.0  0.1   0:00.64 /usr/sbin/VBoxSer
3133 user    20   0  529M  7620 1912  S   0.0  0.7   0:00.21 nm-applet
1507 root     20   0 2044M 2292 1204  S   0.0  0.2   0:00.23 /usr/sbin/console
2024 rtkit    20   0  164M   868   636  S   0.0  0.1   0:00.05 /usr/lib/rtkit/rt
1498 root     20   0  211M   776   444  S   0.2  0.1   0:01.04 /usr/sbin/VBoxSer
1649 root     20   0  337M 22500 14160 S   0.0  2.2   0:00.30 /usr/sbin/apache2
 474 messagebu 20   0 31508 1916   648  S   0.0  0.2   0:00.89 dbus-daemon --sys
1505 root     20   0  211M   776   444  S   0.0  0.1   0:00.24 /usr/sbin/VBoxSer
 581 avahi     20   0 32360   944   628  S   0.0  0.1   0:00.10 avahi-daemon: run
3069 user    20   0 31812 1892   708  S   0.0  0.2   0:00.37 dbus-daemon --for
2980 user    20   0 36348 1392   788  S   0.0  0.1   0:00.09 init --user

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Nice F9Kill F10Quit

```

## Sar (I)

- ★ La herramienta sar (System Activity Report), genera informes de actividad de distintos aspectos del sistema operativo (procesos, cpu, disco...)

```
[root@localhost ~]# sar 1 1
```

```
Linux 5.8.11-1-MANJARO (localhost)      06/10/20      _x86_64_      (8 CPU)
```

16:56:17	CPU	%user	%nice	%system	%iowait	%steal	%idle
16:56:18	all	12,95	0,00	2,05	0,51	0,00	84,49
Media:	all	12,95	0,00	2,05	0,51	0,00	84,49

## Sar (II)

- ★ sar suele estar configurado como actividad de cron:

```
[root@localhost ~]# /etc/cron.d/sysstat
# Run system activity accounting tool every 10 minutes
*/10 * * * * root /usr/lib64/sa/sa1 1 1
# Generate a daily summary of process accounting at 23:53
53 23 * * * root /usr/lib64/sa/sa2 -A
```

- ★ Los resultados se almacenan en /var/log/sa.

```
[root@localhost ~]# ls /var/log/sa
sa01 sa03 sa05 sa07 sa09 sa11 sa13 sa15 sa17 sa19 sa21 sa23 sa25 sa27
sa29 sar01 sar03 sar05 sar08 sar10 sar12 sar14 sar16 sar18 sar20 sar22
sar24 sar26 sar28 sar30
...
```



## Sar (III)

- ★ Si queremos consultar la información del último día usamos **-r**:

```
[root@localhost ~]# sar -r
Linux 5.8.11-1-MANJARO (localhost)      06/10/20      _x86_64_      (8 CPU)
16:56:17      CPU %user      %nice      %system      %iowait      %steal      %idle
16:56:18      all  12,95      0,00      2,05      0,51      0,00      84,49
Media:        all  12,95      0,00      2,05      0,51      0,00      84,49
```

- ★ Para un día concreto usamos la opción **-f**:

```
[root@localhost ~]# sar -S -f /var/log/sa/sa10
Linux 5.8.11-1-MANJARO (localhost)      06/10/20      _x86_64_      (8 CPU)
16:56:17      CPU %user      %nice      %system      %iowait      %steal      %idle
16:56:18      all  12,95      0,00      2,05      0,51      0,00      84,49
Media:        all  12,95      0,00      2,05      0,51      0,00      84,49
```



## Sar (IV)

- ★ Si no indicamos opción alguna, nos muestra los últimos datos almacenados:

```
[root@localhost ~]# sar -r
Linux 5.8.11-1-MANJARO (localhost)      06/10/20      _x86_64_      (8 CPU)
12:20:01 PM   CPU   %user   %nice   %system   %iowait   %steal[...]
12:20:01 PM   all    1.97    0.00     1.84     0.22     0.00[...]
12:30:01 PM   all    0.29    0.00     1.37     0.16     0.00[...] [...]
```

- ★ Al igual que vmstat o iostat, sar permite la toma regular de datos:

```
[root@localhost ~]# sar 1 2
Linux 5.8.11-1-MANJARO (localhost)      06/10/20      _x86_64_      (8 CPU)
06:49:50 PM   CPU   %user   %nice   %system   %iowait   %steal[...]
06:49:51 PM   all    1.05    0.00     6.32     0.00     0.00[...]
Average:       all    0.52    0.00     3.35     0.00     0.00[...]
```

## Sar (V)

- ★ Podemos obtener información sobre la E/S con la opción **-b**:

```
[root@localhost ~]# ssar -b 3 2
Linux 3.10.0-1062.18.1.el7.x86_64 (localhost)      07/10/20      _x86_64_      (4 CPU)
12:14:16      tps      rtps      wtps      bread/s      bwrtn/s
12:14:19      0,33      0,00      0,33      0,00      9,00
12:14:22      0,00      0,00      0,00      0,00      0,00
Media:        0,17      0,00      0,17      0,00      4,50
```

- ★ Con la opción **-q** muestra información sobre colas:

```
[root@localhost ~]# sar -q
Linux 3.10.0-1062.18.1.el7.x86_64 (localhost)      07/10/20      _x86_64_      (4 CPU)
00:00:01      runq-sz  plist-sz  ldavg-1  ldavg-5  ldavg-15  blocked
00:10:01      6       385      0,78     0,99     0,70      0
00:20:01     13      382      0,53     0,67     0,69      0
```

# Mpstat

★ Es similar al top pero orientado a sistemas SMP

```
[root@localhost ~]# mpstat -P ALL
Linux 5.8.11-1-MANJARO (localhost)      06/10/20      _x86_64_      (8 CPU)
17:44:23 CPU %usr  %nice %sys %iowait %irq  %soft %steal %guest %gnice
%idle
17:44:23 all  4,19 0,01 1,15 0,48 0,19 0,12 0,00 0,00 0,00 93,87
17:44:23    0   3,68 0,01 1,32 0,48 0,21 0,10 0,00 0,00 0,00 94,20
17:44:23    1   4,13 0,01 1,16 0,42 0,35 0,45 0,00 0,00 0,00 93,48
17:44:23    2   4,34 0,01 1,13 0,31 0,24 0,10 0,00 0,00 0,00 93,87
17:44:23    3   4,44 0,01 1,11 0,30 0,12 0,06 0,00 0,00 0,00 93,97
17:44:23    4   4,15 0,01 1,10 0,27 0,12 0,06 0,00 0,00 0,00 94,28
17:44:23    5   3,99 0,01 1,22 0,27 0,23 0,05 0,00 0,00 0,00 94,24
17:44:23    6   4,46 0,01 1,08 1,20 0,12 0,06 0,00 0,00 0,00 93,08
17:44:23    7   4,35 0,01 1,07 0,56 0,10 0,04 0,00 0,00 0,00 93,87
```

# iotop

- ★ El comando **iotop** es similar al **top** pero centrado es mostrar información de procesos y sus operaciones de E/S en tiempo real.

```
manav@ubuntu: ~/gfg
Total DISK READ:      0.00 B/s | Total DISK WRITE:      11.25 K/s
Current DISK READ:    0.00 B/s | Current DISK WRITE:      0.00 B/s
  TID  PRIO  USER      DISK READ  DISK WRITE  SWAPIN     IO>     COMMAND
  4108 be/4  manav      0.00 B/s   3.75 K/s   0.00 %    0.00 %  chrome
 11458 be/4  manav      0.00 B/s   3.75 K/s   0.00 %    0.00 %  chrome --type=utilit-es [ThreadPoolForeg]
 10741 be/4  manav      0.00 B/s   0.00 B/s   0.00 %    0.00 %  chrome --type=utilit-es [ThreadPoolForeg]

keys:  any: refresh  q: quit  i: ionice  o: all  p: procs  a: accum
sort:  r: asc  left: SWAPIN  right: COMMAND  home: TID  end: COMMAND
```

## Resumen de herramientas (I)

Utilidad	Descripción	Display Type	Monitoriza
free	Shows the amount of free/used physical and swap memory.	Static	Memory
htop	Enhancement of the top utility, which allows horizontal as well as vertical scrolling, and uses function keys for process control.	Dynamic	CPU Memory Process States Uptime
iftop	Similar to the top utility, it shows current network traffic information, including DNS	Dynamic	Network
iostat	Shows device I/O loading summary broken down per device	Static or Dynamic	CPU Device I/O



## Resumen de herramientas (II)

Utilidad	Descripción	Display Type	Monitoriza
iotp	Similar to the top utility, it shows current I/O usage by processes (or threads).	Dynamic	Device I/O
ip	he -s link option and route option will display network and routing statistics. (Replaces the netstat command.)	Static	NetworkRouting
iptraf	Shows network information, and it is menu driven	Dynamic	Network
Isof	Shows open files and network connections by process.	Static	Network Process map
mpstat	Shows multiple processor statistics.	Static or Dynamic	CPU

## Resumen de herramientas (III)

Utilidad	Descripción	Display Type	Monitoriza
mtr	Shows routing information for the URL parameter	Dynamic	Routing
netstat	The netstat -i option and -r option will display network and routing statistics. This command is considered obsolete. Use ip instead.	Static	Network Routing
ntop	Gathers network statistics that can be viewed via a web browser via port 3000.	Dynamic	Network
pmap	Shows a processes map for the PID parameter.	Static	Process map

## Resumen de herramientas (IV)

Utilidad	Descripción	Display Type	Monitoriza
ps	Shows current process information, including CPU consumption	Static	CPU Process state
ps tree	Shows current processes in a tree format	Static	Process map
sar	Acronym for System Activity Reporter: a multiple resource monitoring utility that collects and displays a wide variety of resource usage information	Static or Dynamic	CPU Memory Network Device I/O
ss	Displays socket statistics directly from kernel space. Provides more information than the netstat utility.	Static	Network

## Resumen de herramientas (V)

Utilidad	Descripción	Display Type	Monitoriza
tcpdump	A packet analyzer/sniffer that shows designated network interface captured packet content descriptions.	Dynamic	Network
top	Multiple display panels that show various resource usage data such as processes consuming the most CPU. Display can easily be changed on the fly. The atop and htop utilities are enhancements of the top command.	Dynamic	CPU Memory Process states Uptime
uptime	Shows how long the system has gone without a reboot, load averages, and current number of users	Static	Uptime



## Resumen de herramientas (VI)

Utilidad	Descripción	Display Type	Monitoriza
vmstat	Shows swap (virtual memory) performance	Static or Dynamic	Memory
w	Shows current user information, including CPU consumption.	Static	CPU Process state

★ Obtenemos un listado de herramientas de monitorización ejecutando:

```
[root@localhost ~]# man -k performance & man -k monitor
```

★ El Display Type lo podemos convertir en Dynamic usando **watch**

```
[root@localhost ~]# watch -n 5 iostat
```



# Resolución de Problemas

---

7

---

# Introducción

- ★ Disponemos de herramientas para medir distintos parámetros de los recursos
- ★ Gracias a estas medidas podemos dar solución a problemas
- ★ Para ello veremos
  - Metodología de resolución
  - Problemas complejos

## Metodologías de resolución

- ★ Aplicamos la siguiente metodología para resolver los problemas:
  - Identificar los síntomas
  - Determinar la raíz del problema
  - Implementar una solución
  - Evaluar los resultados

## Identificar los síntomas

- ★ Un síntoma es aquello por lo que un usuario se queja, no es el problema.
- ★ El problema es aquello que causa el síntoma.
- ★ Debemos identificar, cuantificar y clarificar las condiciones que provocan que aparezca. Objetivo:
  - Ser capaces de reproducir el problema
  - Obtener más información del usuario para identificar la causa raíz del problema

## Determinar la raíz del problema

- ★ Usamos los comandos antes vistos para monitorizar los distintos recursos
  - Herramientas que dan información en tiempo real **vmstat**, **iostat** o **top**
  - Herramientas que recopilan información **sar**.
- ★ Si el problema está relacionado con los recursos
  - Uno o más recursos están usados al 100%. ¿Quién o qué lo provoca?
  - Ninguno está altamente usado: ¿Presentan valores normales de uso?



## Implementar la solución

- ★ No hay una solución única para el problema.
- ★ Hay que intentar buscar la solución que menos afecte al resto del sistema
- ★ Por ejemplo, descubrimos que un proceso ocupa el 100% de la cpu:
  - Solución 1: Matamos el proceso, el usuario pierde los datos
  - Solución 2: Usamos el comando **renice** para reducir la prioridad del proceso

## Evaluar los resultados

- ★ Hay que comprobar que, tras aplicar la solución, el problema desaparece
- ★ Si el problema persiste volveremos a identificar la causa raíz del problema.
- ★ Si la causa está bien identificada, posiblemente la solución esté mal implementada
- ★ Una vez resuelto, determinados si hay que realizar otras acción a medio o largo plazo.

## Problemas complejos

- ★ En ocasión la solución de un problema es compleja ya que los síntomas son causados por una serie de elementos aparentemente no relacionados entre sí.
- ★ Algunos problema conocidos son:
  - La espiral de swap
  - Sin espacio en disco
  - Bloqueo en entrada salida

## La espiral de swap (I)

- ★ Un proceso necesita más memoria que la RAM disponible, usamos la memoria virtual.
- ★ El SO reserva espacio en disco e intercambia páginas residentes en memoria a disco
- ★ El espacio liberado puede ser usado por la aplicación activa
- ★ Cuando las páginas en disco se necesitan (fallo de página), se vuelvan a cargar en memoria y (opcionalmente) se libera espacio (reemplazo de página).
- ★ Cada reemplazo provoca operaciones de E/S de acceso a disco que es mucho más lento que la RAM.



## La espiral de swap (II)

- ★ Si el número de intercambio es pequeño, no lo notaremos
- ★ Si el número es alto (hiperpaginación), el número de operaciones de E/S en el disco se dispara y el sistema no responde.
- ★ Las aplicaciones están bloqueadas a la espera de obtener el recurso que necesitan (una nueva página)
- ★ Si el disco físico usado para swap es el mismo que para datos, el problema se agrava
- ★ Identificamos este problema consultando el estado de actividad de la memoria de swap y consultando la carga media de la cpu (muchos procesos en cola)



## Sin espacio en disco

- ★ Las aplicaciones asumen que todos los accesos a disco se ejecutan sin problemas.
- ★ Si el disco está lleno las aplicaciones entran en un bucle Intentando finalizar las operaciones de escritura en disco con éxito.
- ★ Podemos detectar este comportamiento mediante el comando **strace** que muestra las llamadas al sistemas realizadas por un proceso.

```
[root@localhost ~]# strace ls
execve("/bin/ls", ["ls"], [/ * 21 vars */]) = 0
brk(0)                                = 0x8c31000
access("/etc/ld.so.nohwcap", F_OK)    = -1 ENOENT (No such file or directory)
mmap2(NULL, 8192, PROT_READ, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0xb78c7000
```

## Bloqueo de E/S

- ★ Cuando un proceso realiza una operación de E/S, este se queda bloqueado a la espera de que se satisfaga
- ★ Un proceso bloqueado puede:
  - Ser interrumpido (**S**): operaciones que van a tardar mucho en ejecutarse.
  - No ser interrumpido (**D**): operaciones de acceso al disco

```
[root@localhost ~]# ps aux
apache  26575  0.2 19.6 132572 50104 ?        S    Feb13   3:43 /usr/sbin/httpd
root    8381  57.8  0.2   3844   532 pts/1    D    20:46   0:37 dd
```

- ★ Una carga media de CPU y alta de procesos bloqueados ininterrumpibles indican problemas de E/S