La swap

1. ¿Por qué crear una swap?

En un entorno de 32 bits, un proceso puede acceder teóricamente a 4 GB de espacio de memoria. Dispone de 4 GB de memoria virtual para él de forma exclusiva. Hay varios límites a esta posibilidad:

- El espacio de memoria direccionable de un proceso se comparte entre la zona de código y la zona de datos, cuyo tamaño puede variar según el núcleo utilizado. 3 GB de este espacio se reservan a los datos de proceso, y 1 GB para el núcleo.
- No todos los ordenadores disponen de 4 GB de memoria (aunque sea habitual encontrar servidores de Linux que disponen de 16, 32, 64 GB o incluso más memoria), gracias al núcleo PAE.
- Todos los procesos deben compartir la memoria del ordenador.

En un sistema de 64 bits no hay límite de 4 GB, pero el tamaño de la memoria física disponible sigue siendo fijo.

¿Qué ocurre si un proceso no tiene bastante memoria para tratar sus datos? El sistema operativo descargará segmentos de la memoria física en una zona de intercambio en disco que hará de memoria virtual intermedia. Por lo tanto, hay un intercambio entre la memoria física y esta zona de intercambio, llamada espacio swap. Este proceso permite utilizar más memoria de la que dispone realmente el ordenador, a costa de una importante ralentización del programa si éste resulta ser muy "glotón".

2. Tamaño óptimo

No hay reglas estrictas para establecer el tamaño de la swap. Sin embargo las reglas corrientes siguientes son válidas en la mayoría de casos:

- Si la RAM tiene menos de 512 MB, la swap debe ser el doble de grande.
- Si la RAM tiene entre 1 y 4 GB, la swap debe tener el tamaño de la RAM.

- Si la RAM supera los 4 GB, la swap debe tener 4 GB más o menos, según el uso de los procesos. Estará de acuerdo en que asignar 256 GB de swap a un servidor de 256 GB de RAM no tiene ningún sentido.

Algunos servidores requieren bastante más espacio, como por ejemplo swaps de 8 o 16 GB, o incluso más. En este caso, conviene recurrir a otros modelos de arquitectura hardware o software. En cualquier caso, si su equipo comienza a utilizar el archivo de intercambio de forma permanente y todos los indicadores muestran que el espacio de memoria es pequeño, es hora de añadir memoria física. Un swap en un disco SSD puede mantener una cierta ilusión, pero no es una solución.

Muchos administradores de sistemas eligen no usar swap, prefieren apostar por una cantidad más grande de memoria. De igual forma, algunos editores desaconsejan en algunos casos el uso de swap con algunos productos (openshift, kubernetes, gestión de contenedores).

3. Crear una partición de swap

- Ya sabe crear una partición. Cree una partición con fdisk (o gdisk) del tamaño deseado para la swap y asígnele el tipo **82** (o **8200** para GTP).
- Sincronice la tabla de las particiones con **partprobe**, si fuera necessario.
- Utilice el comando **mkswap** para preparar la partición que se va a recibir de la swap.

```
# mkswap /dev/sdb1
Configurando espacio de intercambio versión 1, tamaño = 32 GiB (34358685696 bytes)
sin etiqueta, UUID=bbaeeae6-fbef-4887-b2d5-32701fc13a44
```

Su swap está lista.

Se puede asignar una etiqueta a la partición de swap con el parámetro `-L`.

Si crea más de 1 o 2 GB de swap, debería pensar en crear varias particiones de swap en discos diferentes ubicados en varios controladores físicos. Linux utilizará cada una de estas particiones, lo que asegura accesos más rápidos.

4. Activar y desactivar la swap

a. Activación dinámica

Linux permite activar y desactivar la swap, o partes de la swap, directamente sin tener que volver a iniciar el sistema.

El comando **swapon** permite activar una partición de swap:

```
# swapon /dev/sda1
```

El parámetro `-p` permite modificar la prioridad de la swap. Cuanto más elevado sea el valor, entre 0 y 32767, más alta será la prioridad de una zona de swap. Linux usa un sistema de prioridades. Este parámetro es útil cuando maneja varias particiones de swap en diferentes discos. En tal caso, dé prioridad al disco más rápido. Si las particiones de swap tienen asignadas las mismas prioridades, Linux repartirá la carga por igual.

Como con mount, el parámetro `-L` permite activar una zona de swap gracias a su etiqueta.

El comando **swapoff** desactiva una zona de swap. Tenga cuidado en disponer del espacio de memoria libre necesario; si no, el comando no funcionará.

El contenido de `/proc/swaps` refleja el estado actual de las zonas de swap activas.

```
# cat /proc/swaps
Filename        Type        Size    Used    Priority
```

```
/dev/sdb1          partition     33553404    0      -2
```

b. En /etc/fstab

Las zonas de swap se colocan en el archivo `/etc/fstab` . Aquí tiene un ejemplo:

```
/dev/sda5 swap swap defaults 0 0
```

Se pueden especificar las opciones **noauto** y **pri=X**. La opción **pri** permite definir la prioridad de la zona de swap.

En el momento de iniciarse, el sistema ejecuta **swapon -a**, que activa todas las particiones de swap presentes en la fstab, excepto si se especifica noauto.

En el momento de la parada, el sistema ejecuta **swapoff -a**, que desactiva completamente la swap.

5. En caso de emergencia: archivo de swap

¿Qué hacer si le falta espacio de swap y ya no es posible crear una nueva partición? Linux sabe utilizar un archivo de intercambio (como Windows, incluso el de Windows). Si queda espacio en uno de sus sistemas de archivos, puede crear encima un archivo de intercambio con un tamaño predefinido. Esta swap será menos eficaz que una partición de swap (problema de fragmentación, tiempo de acceso al sistema de archivos).

Aquí tenemos las operaciones para una pequeña swap de 32 MB:

```
[root@fedora ~]# dd if=/dev/zero of=/swap bs=1024 count=32768
32768+0 registros leídos
32768+0 registros escritos
33554432 bytes (34 MB, 32 MiB) copied, 0,249077 s, 135 MB/s

[root@fedora ~]# mkswap /swap
```

```
mkswap: /swap: permisos 0644 no seguros; se sugiere 0600.
Configurando espacio de intercambio versión 1, tamaño = 32 MiB (33550336 bytes)
sin etiqueta, UUID=e9a8efbc-8401-4784-8866-5e214a59d6f2
[root@fedora ~]# chmod 600 /swap
[root@fedora ~]# sync
[root@fedora ~]# swapon -v /swap

swapon: /swap: firma encontrada [tamaño de página=4096, firma=swap]
swapon: /swap: tamaño de página=4096, tamaño de intercambio=33554432,
tamaño de dispositivo=33554432
swapon /swap
```

Modifique si es preciso el archivo **`/etc/fstab`** para activar esta swap después del montaje de los sistemas de archivos. Se activa la swap en el boot, en general después del montaje de /. Si la nueva swap se encuentra en otra parte (otro punto de montaje), obtendrá un mensaje de error porque la swap ha sido activada antes que los otros puntos de montaje. Por lo tanto, es preferible crear el archivo en el sistema de archivos raíz /.

```
/swap swap swap defaults 0 0
```

6. Estado de la memoria

a. free

El comando **free** le proporciona la información relativa a la memoria física (RAM) de su ordenador, así como a la ocupación de la swap. La unidad por defecto es el KB (idéntico con el parámetro `-k`), pero se puede cambiar a MB (`-m`) e incluso a GB (`-g`).

```
# free -k
      total        used        free      shared  buff/cache   available
Mem:  4024600     1667436      705828       78068     1651336     2020292
Swap: 2117628           0     2117628
```

Sin embargo, tenga cuidado en interpretar correctamente las columnas. En el resultado anterior, aun disponiendo de 2 GB de memoria, el sistema indica que sólo hay libres 650 MB. Esto se debe a que Linux tiende a reservarse todo el espacio disponible en forma de buffers (memoria intermedia) y de caché. El contenido de la caché es volátil; por lo tanto Linux puede liberar en gran medida este espacio para asignarlo a los programas y datos. También es el caso para los buffers.

No obstante, observe la columna **available**. Esta indica que, de hecho, casi 2 GB están disponibles. Las partes del caché no usadas se pueden liberar cuando sea necesario.

El parámetro **-m** muestra el resultado en MB, mientras que **-g** lo muestra en GB y **-h** en "human readable", es decir, de forma legible. El parámetro **-t** muestra el total por cada columna.

```
# free -ht
      total       used       free     shared  buff/cache  available
Mem:   3,8Gi      1,6Gi      677Mi       76Mi      1,6Gi      1,9Gi
Swap:  2,0Gi        0B       2,0Gi
Total: 5,9Gi      1,6Gi      2,7Gi
Highmem, lowmem
```

Highmem, lowmem

Si trabaja en un entorno de 32 bits, un parámetro muy interesante es `-l`, que muestra el detalle de la ocupación de memoria de zonas de memoria alta (highmem) y baja (lowmem). En estos sistemas, la memoria se descompone en las zonas baja y alta. La zona highmem es la zona en la que los procesos tienen su código y sus datos. La zona lowmem es una zona reservada al núcleo y que sirve, entre otras funciones, para almacenar diversas cachés y buffers, así como las tablas de asignación de memoria, para cada proceso. Esta memoria puede asignarse también a los procesos del núcleo. Si un proceso reserva memoria, los datos relativos a su posición y su tamaño reales están en lowmem. De igual modo, todas las transferencias de datos (entrada y salida de disco, por ejemplo) sólo las puede hacer el núcleo y, por lo tanto, estos datos deben transitar de la zona highmem a la zona lowmem.

Esta diferenciación puede conducir a casos sorprendentes donde el sistema operativo puede necesitar memoria y desencadenar mecanismos de autodefensa (matando procesos) mientras que la memoria no parece saturada y el swap no está en uso. El

comando free muestra por defecto la suma de ambas zonas de memoria. Lo que sucede realmente es que la zona lowmem está llena. El siguiente capítulo le mostrará cómo reducir los riesgos de que esto se produzca.

```
$ free -l
             total       used       free     shared    buffers     cached
Mem:       1025488     735324     290164          0      32868     419224
Low:        886408     596528     289880
High:       139080     138796        284
-/+ buffers/cache:     283232     742256
Swap:      1999868          0    1999868
```

Los sistemas en 64 bits no tienen este problema, ya que pueden acceder a toda la memoria, que es de tipo lowmem.

b. Memoria reservada

Una parte de la memoria se reserva para el núcleo del sistema y el resto de los binarios no la pueden utilizar. Es útil para los tratamientos del núcleo, su carga, el initrd. Vea el resultado del comando siguiente (truncado de manera voluntaria):

```
# dmesg | grep -i memory
[    0.344052] Memory: 3987020K/4193848K available (14339K kernel code, 2270K rwdata,
4720K rodata, 2544K init, 5404K bss, 206828K reserved, 0K cma-reserved)
...
[    0.372473] Freeing SMP alternatives memory: 36K
[    0.492362] x86/mm: Memory block size: 128MB
[    1.177904] Freeing initrd memory: 29520K
[    1.191140] Non-volatile memory driver v1.3
[    1.831103] Freeing unused decrypted memory: 2040K
[    1.831460] Freeing unused kernel image memory: 2544K
[    1.835609] Freeing unused kernel image memory: 2016K
[    1.835865] Freeing unused kernel image memory: 1424K
```

El sistema se reserva unos 200 MB de memoria, luego libera la memoria que ya no

necesita para obtener el resultado esperado.

c. meminfo

El sistema de archivos virtual /proc contiene información detallada sobre la memoria mediante el seudoarchivo **`/proc/meminfo`** . Parece bastante difícil encontrar algo más completo. La salida siguiente es el resultado en un sistema Linux de 64 bits. En 32 bits, dos líneas adicionales (highmem y lowmem) indican las zonas reservadas a los datos y al núcleo. Las primeras líneas y las relativas a la swap son idénticas al resultado del comando **free**.

```
# cat /proc/meminfo
MemTotal:       4024600 kB
MemFree:        1182848 kB
MemAvailable:   2398352 kB
Buffers:           1100 kB
Cached:         1440136 kB
SwapCached:           0 kB
Active:         1436560 kB
Inactive:       1021936 kB
Active(anon):   1019484 kB
Inactive(anon):   75808 kB
Active(file):    417076 kB
Inactive(file):  946128 kB
Unevictable:          0 kB
Mlocked:              0 kB
SwapTotal:      2117628 kB
SwapFree:       2117628 kB
Dirty:                0 kB
Writeback: kB
AnonPages:      1017296 kB
Mapped:          538840 kB
Shmem:            78032 kB
KReclaimable:    110772 kB
...
```

d. swap usado y memoria libre

Considere el ejemplo siguiente, basado en una situación real:

```
$ free
              total        used        free      shared  buff/cache   available
Mem:       16268680     4254628     1048788         180    10965264    11706336
Swap:       4194300      154768     4039532
```

Tiene aquí una situación muy extraña: en efecto, el sistema dispone de 1 GB libre, por lo que 150 MB de swap se han consumido. La caché parece utilizar en torno a 10 GB. ¿Por qué este swap?

Linux favorece la caché, colocando en swap los procesos y datos no utilizados. De esta forma, nos podemos encontrar con al parecer poca memoria libre y el swap utilizado. En este caso, la última columna muestra su uso. Pero ¿cómo probar que la memoria no está llena? Vaciando las cachés. Es el momento de hablar de la línea de comandos mágica que efectúa esta acción:

```
# sync ; echo 3 >/proc/sys/vm/drop_caches
# free
              total        used        free      shared  buff/cache   available
Mem:       16268680     4884188    11213772         180      170720    11187496
Swap:       4194300      154768     4039532
```

¿No es impresionante? Continúe vaciando la swap:

```
# swapoff -a
# swapon -a
# free
              total        used        free      shared  buff/cache   available
Mem:       16268680     5038892    11045284         644      184504    11025788
Swap:       4194300           0     4194300
```

Acaba de tranquilizarse, y de tranquilizar a su cliente: no, la memoria no está llena. Al contrario, acaba de, con toda probabilidad, reducir el rendimiento de su servidor: la caché requerirá muchas E/S y acceso a memoria para ser reconstruida.