Beginners guide on linux memory management

Entre los distintos procesos

El artículo aclara que el tema del manejo de memoria en Linux es tan basto que no podría cubrirse en su totalidad en un sólo artículo, pero que desglosaremos el tema en áreas importantes con una vista general de cada una.

Vista general del manejo de memoria en Linux

La CPU (AKA microprocesador o procesador) es la parte central de la computadora y la RAM es el portal que da desde el “front-end” hacia la CPU, todo lo que llega a la CPU pasa por la RAM, pero una vez que una serie de procesos llegan al CPU este los va “enfilando” construyendo su caché, que son pocas pues son caras y poco útiles para algunas de las instrucciones pero funcionan como “pequeñas memorias RAM” para la CPU.

Al hacer un llamado al disco duro, este le pasa lo solicitado a la RAM creando una copia con la que el usuario va a estar trabajando, al hacer cambios en este se pasarán a un buffer que servirá para almacenar cambios recientes, una vez que estos cambios hayan pasado el suficiente tiempo en la RAM pasarán al disco duro.

En general, todo lo que ocurre dentro de una computadora pasa a través de la RAM, la RAM es esencial.

Entendiendo la memoria virtual

Linux trabaja con una memoria virtual y una “residente”. Esta primera memoria es literal una memoria que no existe a la que el kernel puede acceder.

La memoria virtual es usada por el kernel de Linux para reservar memoria para un programa en específico para después no dejar que ningún otro programa intente reservar el mismo espacio.

Cuando el programa va a usar su espacio de memoria reservada, lo que significa que ese espacio va a ser alojado por tanto pasa a la memoria “residente”. Pero si la memoria “residente” comienza a agotarse (OOM - Out Of Memory), el “asesino de OOM” entra en acción y “mata” de manera casi aleatoria algún proceso que no “merezca” usar tanto la memoria residente.

Page Cache

Retomando el tema de la RAM, cuando esta “jala” datos del disco duro, estos se almacenan en una parte de la memoria (RAM) a la que se le llama “pagecache”, la caché es una parte de la memoria que almacena datos para tener acceso a ellos de una manera más rápida eficientando el rendimiento I/O (E/S en español).

¿Qué hay del buffer?
El buffer es la representación del bloque de disco de los datos que se encuentran “debajo” de la “pagecache”, son metadatos de los archivos o datos que residen en esta última. Es decir, siempre que se quiere acceder a datos de la “pagecahe” primero se fija en el buffer para obtener los metadatos.

Lectura y escritura

Leyendo desde el disco

Se hace referencia al “pagecache” para agregar páginas satisfaciendo la petición de lectura, si alguna página aún no está siendo usada se agrega con los datos leídos, si hay suficiente memoria, la página se queda almacenada por tiempo indefinido y puede ser reutilizada por otros procesos sin que tengan que leerlos directamente del disco.

Escribiendo en el disco

De igual forma se hace referencia al “pagecache” para corroborar que lo que se va a escribir en disco ya está ahí, si no se agrega. A la hora de la escritura no se hace inmediatamente, se deja un tiempo para modificar estos datos y después los escribe el Kernel.

Páginas sucias

Las páginas sucias son páginas del “pagecache” que son modificadas por algún proceso, la idea es que estas permanezcan la mayor cantidad posible en la memoria, pero para evitar su pérdida el Kernel las escribe en el disco cuando la “pagecache” ya está llena o si ha pasado mucho tiempo desde la última modificación que se le hizo a los datos por algún proceso.

Esta escritura en el disco de las “páginas sucias” se puede dividir en tres escenarios:

Periódico

Los hilos encargados de hacer la escritura son despertados de manera periódica.

Segundo plano (background)

El kernel escribe la mayor cantidad de páginas sucias como le sea posible o al menos hasta estar por debajo de los límites, esto lo hace en “segundo plano” o “de fondo”, así no interrumpiendo otros procesos.

Activo

Se bloquean las tareas que generan “páginas sucias” para evitar que se alcance el límite basado en los parámetros de “*dirty_**”.

TLB

Los Buffers de traducción a los lados o lateral (Translation Lookaside Buffers) son usados para ubicar las páginas que están referidas en la memoria virtual. Estos buffers pueden estar en memoria física, si lo están este proceso puede ser muy rápido. Existen dos escenarios importantes aquí:

TLB miss AKA minor page fault

Es cuando se lanza un recorrido para ubicar a la página de memoria y cargarse. Conocido como “error de página *menor*”.

Major page fault

Ocurre cuando es necesario recuperar una página de memoria del swap (lo que esto quiera significar).

Memoria activa e inactiva

En resumen, la memoria activa es aquella que se está utilizando para algo, mientras la inactiva es aquella que no.

Diferentes tipos de *swapping* y sus riesgos

En esencia, si la cantidad de “swap” que es usada supera al tamaño de páginas de “memoria anónima” (esto puede verse en el archivo `/proc/meminfo`) significa que la memoria activa ha sido “swappeada” lo que se traduce en mal rendimiento y en que necesitas instalar más RAM pues hay mucho tráfico de E/S que alentará tu equipo.