第 0019 讲 14slab 分配器案例实战分析 2

一、slab 的原理机制

Linux 内核中的 slab 是一种高效、灵活、可扩展的内存管理算法，它能够提供快速而稳定的内存分配服务。在实际应用中，slab 已经成为 Linux 内核中最常用的内存分配机制之一。

内存分配算法主要原理如下：

- slab 将内存按照大小划分成若干个块，每个块称为一个 slab。不同大小的对象可以被放在不同大小的 slab 中。
- 当需要分配内存时，slab 会首先查找是否有可用的 slab 来满足请求。如果有，则从该 slab 中取出空闲对象并返回；如果没有，则新建一个 slab，并将其中的对象标记为已使用状态。
- 当释放一个对象时，slab 会将其重新标记为未使用状态，并且可以选择将该对象归还到对应的 slab 中以备下次使用。
- slab 还支持缓存和预先分配功能。当创建新的 slab 时，可以预先将一定数量对象加入到该 slab 中以减少后续空间碎片问题。同时，在频繁使用某些类型的对象时，也可以通过缓存机制来提高访问速度。

- 由于不同类型的对象可能具有不同生命周期和频率，因此 slab 允许动态增长和收缩。当需要更多空间时，可以动态增加新的 slab；当空闲空间过多时，也可以回收一些不再使用的 slab 来节约内存。

二、slab 应用场景

- 文件系统；
- 内核模块；
- 驱动程序；
- 网络协议栈。

三、实战案例分析

```
root@ubuntu:/home/vico/Desktop/slab# make
make -C /usr/src/linux-headers-5.4.0-150-generic    M=/home/vico/Desktop/slab
modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-150-generic'
  Building modules, stage 2.
  MODPOST 1 modules
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-150-generic'
root@ubuntu:/home/vico/Desktop/slab# ls -l
total 40
-rwxrw-rw- 1 vico vico  277 Jul 10 23:37 Makefile
-rw-r--r-- 1 root root   32 Jul 11 01:40 modules.order
-rw-r--r-- 1 root root    0 Jul 10 23:37 Module.symvers
-rw------- 1 vico vico 2718 Jul 11 00:37 slab.c
-rw-r--r-- 1 root root 6872 Jul 11 00:37 slab.ko
-rw-r--r-- 1 root root   32 Jul 11 00:37 slab.mod
-rw-r--r-- 1 root root  924 Jul 11 00:37 slab.mod.c
-rw-r--r-- 1 root root 3232 Jul 11 00:37 slab.mod.o
-rw-r--r-- 1 root root 4152 Jul 11 00:37 slab.o
root@ubuntu:/home/vico/Desktop/slab# insmod slab.ko
root@ubuntu:/home/vico/Desktop/slab# dmesg -c
[ 7850.725605] Prompt：Create MyCacheTest successfully.
[ 7850.725629] Prompt：successfully created a object, kmembuffer_address=0x0000
000068b0df9e
root@ubuntu:/home/vico/Desktop/slab# rmmod slab.ko
root@ubuntu:/home/vico/Desktop/slab# dmesg -c
[ 7864.910912] Prompt：destroyed a cache object.
[ 7864.924359] Prompt：destroyed MyCacheTest.
root@ubuntu:/home/vico/Desktop/slab#
```

零声教育-Linux内核源码分析课程