理解和配置 Linux 下的 OOM Killer

2013年10月10日 | 标签: linux kernel, mysql, oom killer | 作者：vpsee

最近有位 VPS 客户抱怨 MySQL 无缘无故挂掉，还有位客户抱怨 VPS 经常死机，登陆到终端看了一下，都是常见的 Out of memory 问题。这通常是因为某时刻应用程序大量请求内存导致系统内存不足造成的，这通常会触发 Linux 内核里的 Out of Memory (OOM) killer，OOM killer 会杀掉某个进程以腾出内存留给系统用，不致于让系统立刻崩溃。如果检查相关的日志文件（/var/log/messages）就会看到下面类似的 Out of memory: Kill process 信息：

```
...
Out of memory: Kill process 9682 (mysqld) score 9 or sacrifice child
Killed process 9682, UID 27, (mysqld) total-vm:47388kB, anon-rss:3744kB, file-rss:80kB
httpd invoked oom-killer: gfp_mask=0x201da, order=0, oom_adj=0, oom_score_adj=0
httpd cpuset=/ mems_allowed=0
Pid: 8911, comm: httpd Not tainted 2.6.32-279.1.1.el6.i686 #1
...
21556 total pagecache pages
21049 pages in swap cache
Swap cache stats: add 12819103, delete 12798054, find 3188096/4634617
Free swap  = 0kB
Total swap = 524280kB
131071 pages RAM
0 pages HighMem
3673 pages reserved
67960 pages shared
124940 pages non-shared
```

Linux 内核根据应用程序的要求分配内存，通常来说应用程序分配了内存但是并没有实际全部使用，为了提高性能，这部分没用的内存可以留作它用，这部分内存是属于每个进程的，内核直接回收利用的话比较麻烦，所以内核采用一种过度分配内存（over-commit memory）的办法来间接利用这部分“空闲”的内存，提高整体内存的使用效率。一般来说这样做没有问题，但当大多数应用程序都消耗完自己的内存的时候麻烦就来了，因为这些应用程序的内存需求加起来超出了物理内存（包括 swap）的容量，内核（OOM killer）必须杀掉一些进程才能腾出空间保障系统正常运行。用银行的例子来讲可能更容易懂一些，部分人取钱的时候银行不怕，银行有足够的存款应付，当全国人民（或者绝大多数）都取钱而且每个人都想把自己钱取完的时候银行的麻烦就来了，银行实际上是没有这么多钱给大家取的。

内核检测到系统内存不足、挑选并杀掉某个进程的过程可以参考内核源代码 linux/mm/oom_kill.c，当系统内存不足的时候，out_of_memory() 被触发，然后调用 select_bad_process() 选择一个“bad”进程杀掉，如何判断和选择一个“bad”进程呢，总不能随机选吧？挑选的过程由 oom_badness() 决定，挑选的算法和想法都很简单很朴实：最 bad 的那个进程就是那个最占用内存的进程。

```
/**
 * oom_badness - heuristic function to determine which candidate task to kill
 * @p: task struct of which task we should calculate
 * @totalpages: total present RAM allowed for page allocation
 *
 * The heuristic for determining which task to kill is made to be as simple and
 * predictable as possible.  The goal is to return the highest value for the
 * task consuming the most memory to avoid subsequent oom failures.
 */
unsigned long oom_badness(struct task_struct *p, struct mem_cgroup *memcg,
      const nodemask_t *nodemask, unsigned long totalpages)
{
 long points;
 long adj;

 if (oom_unkillable_task(p, memcg, nodemask))
  return 0;

 p = find_lock_task_mm(p);
 if (!p)
  return 0;

 adj = (long)p->signal->oom_score_adj;
 if (adj == OOM_SCORE_ADJ_MIN) {
  task_unlock(p);
  return 0;
 }

 /*
  * The baseline for the badness score is the proportion of RAM that each
  * task's rss, pagetable and swap space use.
  */
 points = get_mm_rss(p->mm) + p->mm->nr_ptes +
   get_mm_counter(p->mm, MM_SWAPENTS);
```

```
 task_unlock(p);

 /*
  * Root processes get 3% bonus, just like the __vm_enough_memory()
  * implementation used by LSMs.
  */
 if (has_capability_noaudit(p, CAP_SYS_ADMIN))
  adj -= 30;

 /* Normalize to oom_score_adj units */
 adj *= totalpages / 1000;
 points += adj;

 /*
  * Never return 0 for an eligible task regardless of the root bonus and
  * oom_score_adj (oom_score_adj can't be OOM_SCORE_ADJ_MIN here).
  */
 return points > 0 ? points : 1;
}
```

上面代码里的注释写的很明白，理解了这个算法我们就理解了为啥 MySQL 躺着也能中枪了，因为它的体积总是最大（一般来说它在系统上占用内存最多），所以如果 Out of Memeory (OOM) 的话总是不幸第一个被 kill 掉。解决这个问题最简单的办法就是增加内存，或者想办法优化 MySQL 使其占用更少的内存，除了优化 MySQL 外还可以优化系统（优化 Debian 5，优化 CentOS 5.x），让系统尽可能使用少的内存以便应用程序（如 MySQL）能使用更多的内存，还有一个临时的办法就是调整内核参数，让 MySQL 进程不容易被 OOM killer 发现。

配置 OOM killer

我们可以通过一些内核参数来调整 OOM killer 的行为，避免系统在那里不停的杀进程。比如我们可以在触发 OOM 后立刻触发 kernel panic，kernel panic 10秒后自动重启系统。

```
# sysctl -w vm.panic_on_oom=1
vm.panic_on_oom = 1

# sysctl -w kernel.panic=10
kernel.panic = 10

# echo "vm.panic_on_oom=1" >> /etc/sysctl.conf
# echo "kernel.panic=10" >> /etc/sysctl.conf
```

从上面的 oom_kill.c 代码里可以看到 oom_badness() 给每个进程打分，根据 points 的高低来决定杀哪个进程，这个 points 可以根据 adj 调节，root 权限的进程通常被认为很重要，不应该被轻易杀掉，所以打分的时候可以得到 3% 的优惠（adj -= 30; 分数越低越不容易被杀掉）。我们可以在用户空间通过操作每个进程的 oom_adj 内核参数来决定哪些进程不这么容易被 OOM killer 选中杀掉。比如，如果不想 MySQL 进程被轻易杀掉的话可以找到 MySQL 运行的进程号后，调整 oom_score_adj 为 -15（注意 points 越小越不容易被杀）：

```
# ps aux | grep mysqld
mysql    2196  1.6  2.1 623800 44876 ?        Ssl  09:42   0:00 /usr/sbin/mysqld

# cat /proc/2196/oom_score_adj
0
# echo -15 > /proc/2196/oom_score_adj
```

当然，如果需要的话可以完全关闭 OOM killer（不推荐用在生产环境）：

```
# sysctl -w vm.overcommit_memory=2

# echo "vm.overcommit_memory=2" >> /etc/sysctl.conf
```

找出最有可能被 OOM Killer 杀掉的进程

我们知道了在用户空间可以通过操作每个进程的 oom_adj 内核参数来调整进程的分数，这个分数也可以通过 oom_score 这个内核参数看到，比如查看进程号为981的 omm_score，这个分数被上面提到的 omm_score_adj 参数调整后（ - 15），就变成了3：

```
# cat /proc/981/oom_score
18

# echo -15 > /proc/981/oom_score_adj
# cat /proc/981/oom_score
3
```

也越来

路过：这个帖子 好多年啦 评论读都读不完 就弱弱的问一句 有人用 debian 的吗

fans：就是因为上面那些人不负责任的吹牛逼，列举大量mac的优势，实际上上手才知道，根本，没什么卵用，才会使得更多无知的人莫名其妙的入了mac，结果发现什么都做不了，m

sshfs：这个需要mark一下

gghdf：mac挺难用的

：大赞，困扰了好久，惬意，无与伦比的惬意！！！

忘川小路：ldap 没有active这个属性，是通过userAccountControl属性控制的

：很底层的东西，厉害了，我的博主！

NCTUNS 6.0：HI i am PhD student and i need NCTuns 6.0 version. please send me via my emai

友链

Fwolf's Blog

I am LAZY bones

LinuxTOY

NoSQLFan

OwnLinux

小明明

笑遍世界

关于

关于

购买 VPS

订阅 RSS

下面这个 bash 脚本可用来打印当前系统上 oom_score 分数最高 (最容易被 OOM Killer 杀掉) 的进程 :

```
# vi oomscore.sh
#!/bin/bash
for proc in $(find /proc -maxdepth 1 -regex '/proc/[0-9]+'); do
    printf "%2d %5d %s\n" \
        "$(cat $proc/oom_score)" \
        "$(basename $proc)" \
        "$(cat $proc/cmdline | tr '\0' ' ' | head -c 50)"
done 2>/dev/null | sort -nr | head -n 10

# chmod +x oomscore.sh
# ./oomscore.sh
18   981 /usr/sbin/mysqld
 4 31359 -bash
 4 31056 -bash
 1 31358 sshd: root@pts/6
 1 31244 sshd: vpsee [priv]
 1 31159 -bash
 1 31158 sudo -i
 1 31055 sshd: root@pts/3
 1 30912 sshd: vpsee [priv]
 1 29547 /usr/sbin/sshd -D
```

分类 : Linux | BSD | Solaris　　发表评论(15 Comments)

评论 (15 Comments)

1. **rooter** - October 10th, 2013 10:41 pm
   这个很深，博主功力深厚！佩服！

2. **豆沙包** - October 11th, 2013 3:43 am
   赞！成功帮我解决了问题，谢谢！

3. **SR1** - October 11th, 2013 3:47 am
   原来一个简单的OOM背后都有这么大的学问！又长知识了。

4. **waveacme** - October 15th, 2013 1:03 pm
   讲解的很清楚，不错。

5. **秦腔** - October 15th, 2013 4:52 pm
   上次我的vps也这样挂掉的。。可惜那时没看到你这文章，折腾了老久。。

6. **zizizhe** - January 10th, 2014 6:15 pm
   有价值的文章，理解了流程，受益匪浅

7. **e** - March 6th, 2015 11:45 am
   liuB

8. **刀尖红叶** - June 15th, 2015 5:24 pm
   感谢分享！

9. **大麦哲伦星系学家** - September 23rd, 2015 11:29 am
   大赞，很深刻的讲解，完全理解了OOM killer的机制以及处理办法，帮我解决了问题

10. **异步人生** - May 9th, 2016 7:43 pm
    博主功力深厚，我刚刚学linux，以后会经常一直关注博主的文章的。

11. **博主功力深厚** - January 2nd, 2017 5:01 pm
    看了博主的文章，感觉遇到了真高人。

12. **Anonymous** - April 5th, 2017 4:46 pm
    经典。

13. **Anonymous** - July 24th, 2017 4:56 pm
    oom_adj相同的话先杀哪个进程呢？

14. **刘权** - July 26th, 2017 4:52 pm
    大牛

15. **Anonymous** - August 16th, 2017 6:08 pm
    很底层的东西，厉害了，我的博主！

发表评论

昵称：

□□