使用 nice、cpulimit 和 cgroups 限制 cpu 占用率

编译自：<http://blog.scoutapp.com/articles/2014/11/04/restricting-process-cpu-usage-using-nice-cpulimit-and-cgroups>  
本文地址：<https://linux.cn/article-4742-1.html>

2015-01-26 07:42    评论: [19](https://linux.cn/portal.php?mod=comment&id=4742&idtype=aid) 收藏: 6 分享: 41



Linux内核是一名了不起的马戏表演者，它在进程和系统资源间小心地玩着杂耍，并保持系统的正常运转。 同时，内核也很公正：它将资源公平地分配给各个进程。

但是，如果你需要给一个重要进程提高优先级时，该怎么做呢？ 或者是，如何降低一个进程的优先级？ 又或者，如何限制一组进程所使用的资源呢？

答案是需要由用户来为内核指定进程的优先级

大部分进程启动时的优先级是相同的，因此Linux内核会公平地进行调度。 如果想让一个CPU密集型的进程运行在较低优先级，那么你就得事先配置好调度器。

下面介绍3种控制进程运行时间的方法：

* 使用 nice 命令手动降低任务的优先级。
* 使用 cpulimit 命令不断的暂停进程，以控制进程所占用处理能力不超过特定限制。
* 使用linux内建的control groups（控制组）功能，它提供了限制进程资源消耗的机制。

我们来看一下这3个工具的工作原理和各自的优缺点。

模拟高cpu占用率

在分析这3种技术前，我们要先安装一个工具来模拟高CPU占用率的场景。我们会用到CentOS作为测试系统，并使用[Mathomatic toolkit](http://www.mathomatic.org/)中的质数生成器来模拟CPU负载。

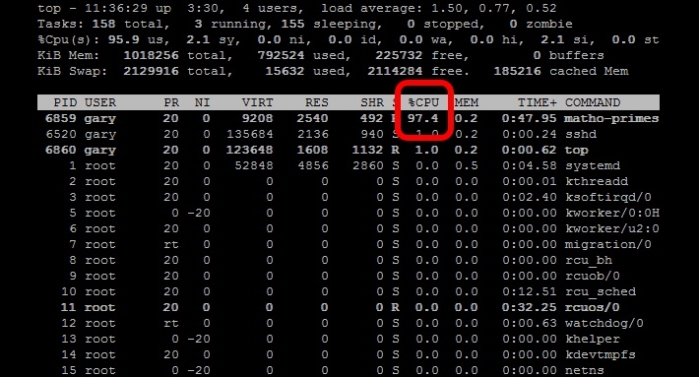
很不幸，在CentOS上这个工具没有预编译好的版本，所以必须要从源码进行安装。先从 <http://mathomatic.orgserve.de/mathomatic-16.0.5.tar.bz2> 这个链接下载源码包并解压。然后进入 mathomatic-16.0.5/primes 文件夹，运行 make 和 sudo make install 进行编译和安装。这样，就把 matho-primes 程序安装到了 /usr/local/bin 目录中。

接下来，通过命令行运行：

1. /usr/local/bin/matho-primes 0 9999999999 > /dev/null &

程序运行后，将输出从0到9999999999之间的质数。因为我们并不需要这些输出结果，直接将输出重定向到/dev/null就好。

现在，使用top命令就可以看到matho-primes进程榨干了你所有的cpu资源。



好了，接下来（按q键）退出 top 并杀掉 matho-primes 进程（使用 fg 命令将进程切换到前台，再按 CTRL+C）

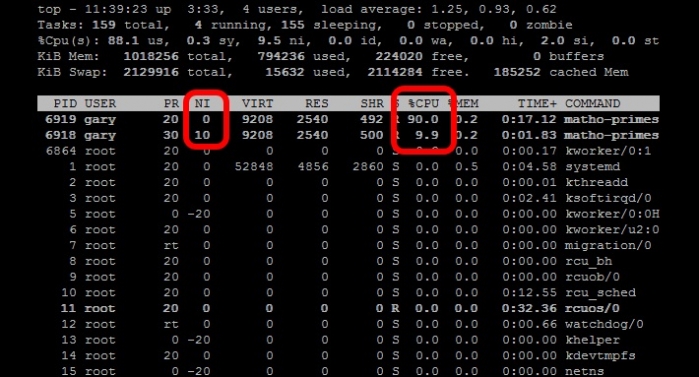
nice命令

下面介绍一下nice命令的使用方法，nice命令可以修改进程的优先级，这样就可以让进程运行得不那么频繁。 这个功能在运行cpu密集型的后台进程或批处理作业时尤为有用。 nice值的取值范围是[-20,19],-20表示最高优先级，而19表示最低优先级。 Linux进程的默认nice值为0。使用nice命令（不带任何参数时）可以将进程的nice值设置为10。这样调度器就会将此进程视为较低优先级的进程，从而减少cpu资源的分配。

下面来看一个例子，我们同时运行两个 matho-primes 进程，一个使用nice命令来启动运行，而另一个正常启动运行：

1. nice matho-primes 0 9999999999 > /dev/null &
2. matho-primes 0 9999999999 > /dev/null &

再运行top命令。



看到没，正常运行的进程（nice值为0）获得了更多的cpu运行时间，相反的，用nice命令运行的进程占用的cpu时间会较少（nice值为10）。

在实际使用中，如果你要运行一个CPU密集型的程序，那么最好用nice命令来启动它，这样就可以保证其他进程获得更高的优先级。 也就是说，即使你的服务器或者台式机在重载的情况下，也可以快速响应。

nice 还有一个关联命令叫做 renice，它可以在运行时调整进程的 nice 值。使用 renice 命令时，要先找出进程的 PID。下面是一个例子：

1. renice +10 1234

其中，1234是进程的 PID。

测试完 nice 和 renice 命令后，记得要将 matho-primes 进程全部杀掉。

cpulimit命令

接下来介绍 cpulimit 命令的用法。 cpulimit 命令的工作原理是为进程预设一个 cpu 占用率门限，并实时监控进程是否超出此门限，若超出则让该进程暂停运行一段时间。cpulimit 使用 SIGSTOP 和 SIGCONT 这两个信号来控制进程。它不会修改进程的 nice 值，而是通过监控进程的 cpu 占用率来做出动态调整。

cpulimit 的优势是可以控制进程的cpu使用率的上限值。但与 nice 相比也有缺点，那就是即使 cpu 是空闲的，进程也不能完全使用整个 cpu 资源。

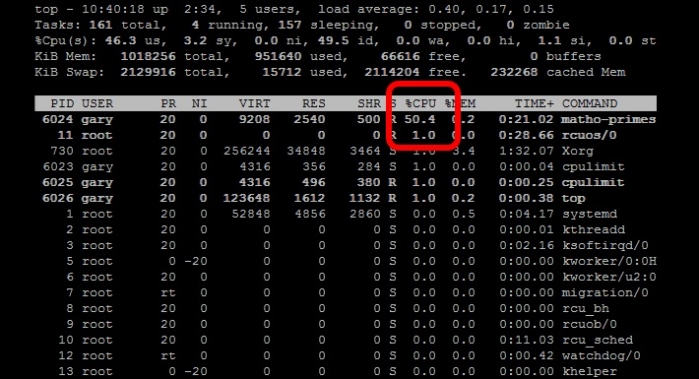
在 CentOS 上，可以用下面的方法来安装它：

1. wget -O cpulimit.zip https:*//github.com/opsengine/cpulimit/archive/master.zip*
2. unzip cpulimit.zip
3. cd cpulimit-master
4. make
5. sudo cp src/cpulimit /usr/bin

上面的命令行，会先从从 GitHub 上将源码下载到本地，然后再解压、编译、并安装到 /usr/bin 目录下。

cpulimit 的使用方式和 nice 命令类似，但是需要用户使用 -l 选项显式地定义进程的 cpu 使用率上限值。举例说明：

1. cpulimit -l 50 matho-primes 0 9999999999 > /dev/null &



从上面的例子可以看出 matho-primes 只使用了50%的 cpu 资源，剩余的 cpu 时间都在 idle。

cpulimit 还可以在运行时对进程进行动态限制，使用 -p 选项来指定进程的 PID，下面是一个实例：

1. cpulimit -l 50 -p 1234

其中，1234是进程的 PID。

cgroups 命令集

最后介绍，功能最为强大的控制组（cgroups）的用法。cgroups 是 Linux 内核提供的一种机制，利用它可以指定一组进程的资源分配。 具体来说，使用 cgroups，用户能够限定一组进程的 cpu 占用率、系统内存消耗、网络带宽，以及这几种资源的组合。

对比nice和cpulimit，cgroups 的优势在于它可以控制一组进程，不像前者仅能控制单进程。同时，nice 和 cpulimit 只能限制 cpu 使用率，而 cgroups 则可以限制其他进程资源的使用。

对 cgroups 善加利用就可以控制好整个子系统的资源消耗。就拿 CoreOS 作为例子，这是一个专为大规模服务器部署而设计的最简化的 Linux 发行版本，它的 upgrade 进程就是使用 cgroups 来管控。这样，系统在下载和安装升级版本时也不会影响到系统的性能。

下面做一下演示，我们将创建两个控制组（cgroups），并对其分配不同的 cpu 资源。这两个控制组分别命名为“cpulimited”和“lesscpulimited”。

使用 cgcreate 命令来创建控制组，如下所示：

1. sudo cgcreate -g cpu:/cpulimited
2. sudo cgcreate -g cpu:/lesscpulimited

其中“-g cpu”选项用于设定 cpu 的使用上限。除 cpu 外，cgroups 还提供 cpuset、memory、blkio 等控制器。cpuset 控制器与 cpu 控制器的不同在于，cpu 控制器只能限制一个 cpu 核的使用率，而 cpuset 可以控制多个 cpu 核。

cpu 控制器中的 cpu.shares 属性用于控制 cpu 使用率。它的默认值是 1024，我们将 lesscpulimited 控制组的 cpu.shares 设为1024（默认值），而 cpulimited 设为512，配置后内核就会按照2：1的比例为这两个控制组分配资源。

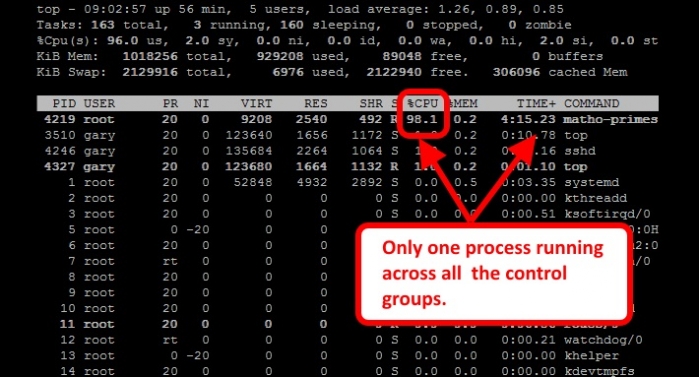
要设置 cpulimited 组的 cpu.shares 为 512，输入以下命令：

1. sudo cgset -r cpu.shares=512 cpulimited

使用 cgexec 命令来启动控制组的运行，为了测试这两个控制组，我们先用cpulimited 控制组来启动 matho-primes 进程，命令行如下：

1. sudo cgexec -g cpu:cpulimited /usr/local/bin/matho-primes 0 9999999999 > /dev/null &

打开 top 可以看到，matho-primes 进程占用了所有的 cpu 资源。



因为只有一个进程在系统中运行，不管将其放到哪个控制组中启动，它都会尽可能多的使用cpu资源。cpu 资源限制只有在两个进程争夺cpu资源时才会生效。

那么，现在我们就启动第二个 matho-primes 进程，这一次我们在 lesscpulimited 控制组中来启动它：

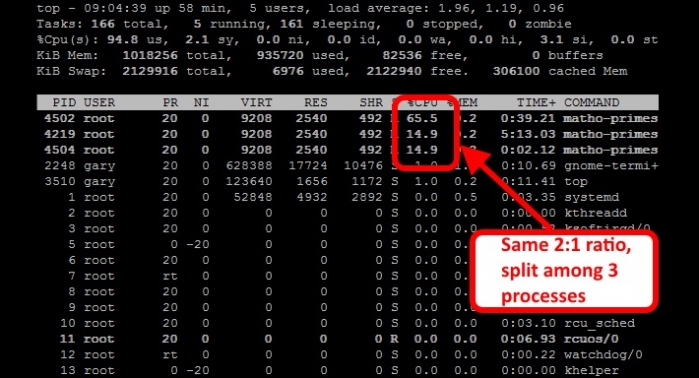
1. sudo cgexec -g cpu:lesscpulimited /usr/local/bin/matho-primes 0 9999999999 > /dev/null &

再打开 top 就可以看到，cpu.shares 值大的控制组会得到更多的 cpu 运行时间。



现在，我们再在 cpulimited 控制组中增加一个 matho-primes 进程：

1. sudo cgexec -g cpu:cpulimited /usr/local/bin/matho-primes 0 9999999999 > /dev/null &

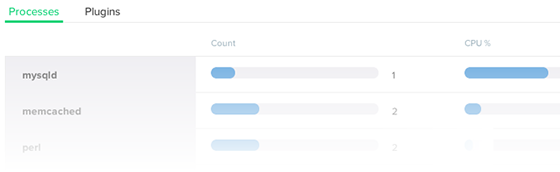


看到没，两个控制组的 cpu 的占用率比例仍然为2：1。其中，cpulimited 控制组中的两个 matho-primes 进程获得的cpu 时间基本相当，而另一组中的 matho-primes 进程显然获得了更多的运行时间。

更多的使用方法，可以在 Red Hat 上查看详细的 cgroups 使用[说明](https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Resource_Management_and_Linux_Containers_Guide/chap-Introduction_to_Control_Groups.html)。（当然CentOS 7也有）

使用Scout来监控cpu占用率

监控cpu占用率最为简单的方法是什么？[Scout](https://scoutapp.com/) 工具能够监控能够自动监控进程的cpu使用率和内存使用情况。



[Scout](https://scoutapp.com/)的触发器（trigger）功能还可以设定 cpu 和内存的使用门限，超出门限时会自动产生报警。

从这里可以获取 [Scout](https://scoutapp.com/) 的试用版。