edsionte's TechBlog No Pains, No Gains

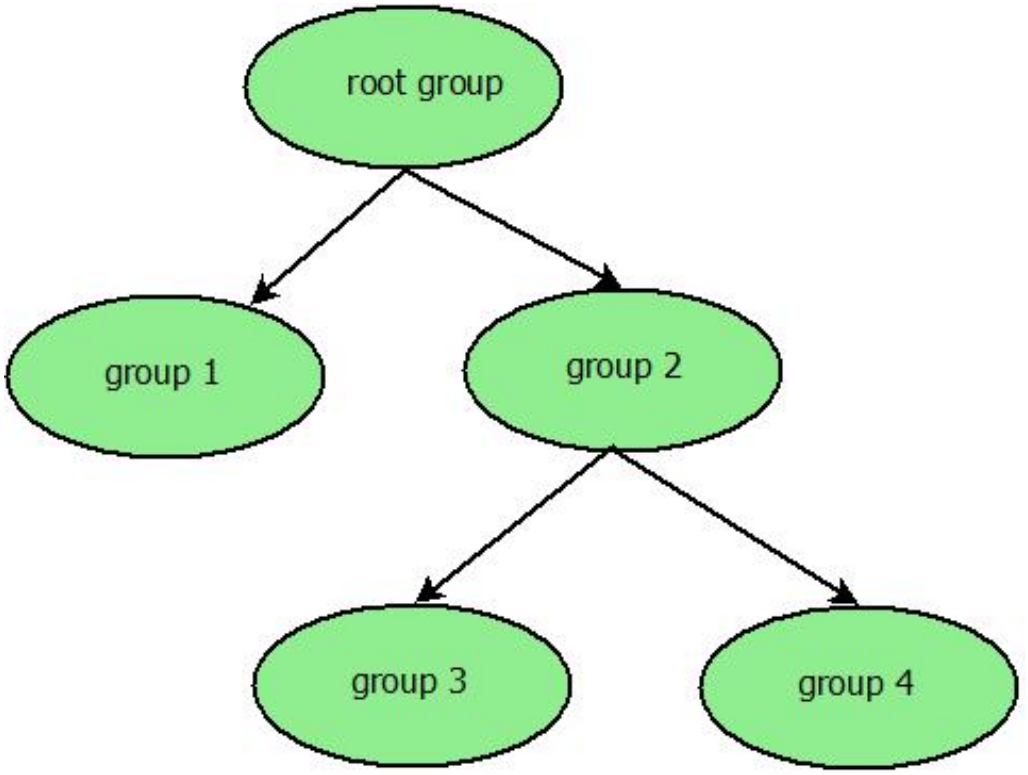
Cgroup简介-概述

2013年3月10日 由 edsionte 留言 »

Cgroup（Control Groups）是这样一种机制：它以分组的形式对进程使用系统资源的行为进行管理和控制。也就是说，用户通过cgroup对所有进程进行分组，再对该分组整体进行资源的分配和控制。

1 Cgroup的结构

cgroup中的每个分组称为进程组，它包含多个进程。最初情况下，系统内的所有进程形成一个进程组（根进程组），根据系统对资源的需求，这个根进程组将被进一步细分为子进程组，子进程组内的进程是根进程组内进程的子集。而这些子进程组很有可能继续被进一步细分，最终，系统内所有的进程组形成一颗具有层次等级（hierarchy）关系的进程组树。如下图：



[http://edsionte.com/techblog/wordpress/wp-content/uploads/2013/03/cgroup_tree.jpeg]

由于进程组可以被进一步划分，因此一个进程可能处于多个进程组中，但这些进程组必然不处于同一层级中。

另外，如果某个进程组内的进程创建了子进程，那么该子进程默认与父进程处于同一进程组中。也就是说，cgroup对改进程组的资源控制同样作用于子进程。

2 subsystem

cgroup是一种对进程资源管理和控制的统一框架，它提供的是一种机制（mechanism），而具体的策略（policy）是通过子系统（subsystem）来完成的，子系统是cgroup对进程组进行资源控制的具体行为。机制和策略是Linux操作系统中一种经典的设计思想，所谓机制就是“我要提供哪种功能”，而策略则是“我要怎样来实现这种功能”。

cgroup中每个子系统都代表一种类型的资源，具体如下：

1) cpu子系统：该子系统为每个进程组设置一个使用CPU的权重值，以此来管理进程对cpu的访问。

2) cpuset子系统：对于多核cpu，该子系统可以设置进程组只能在指定的核上运行，并且还可以设置进程组在指定的内存节点上申请内存。

3) cpuacct子系统：该子系统只用于生成当前进程组内的进程对cpu的使用报告。

4) memory子系统：该子系统提供了以页面为单位对内存的访问，比如对进程组设置内存使用上限等，同时可以生成内存资源报告

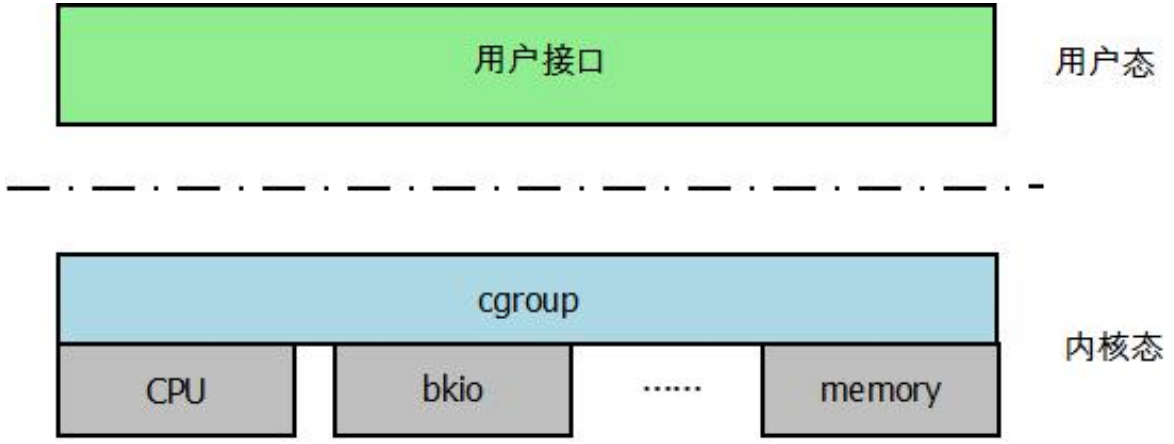
5) blkio子系统：该子系统用于限制每个块设备的输入输出。首先，与CPU子系统类似，该系统通过为每个进程组设置权重来控制块设备对其的I/O时间；其次，该子系统也可以限制进程组的I/O带宽以及IOPS。

6) devices子系统：通过该子系统可以限制进程组对设备的访问，即该允许或禁止进程组对某设备的访问。

7) freezer子系统：该子系统可以使得进程组中的所有进程挂起。

8) net-cls子系统：该子系统提供对网络带宽的访问限制，比如对发送带宽和接收带宽进程限制。

如果要实现子系统对所属进程组的资源控制，那么就要实现该子系统对应的钩子函数。这个关系与虚拟文件系统类似，VFS提供统一的用户接口，而具体的文件操作则通过文件系统（比如ext3）对钩子函数的实现。具体关系如下图：



[http://edsionte.com/techblog/wordpress/wp-content/uploads/2013/03/cgroup_struct.jpeg]

由图可以看出，cgroup在用户态提供统一的用户接口，而每个子系统对资源的控制功能则通过其钩子函数实现。这样使得cgroup在上层是一个统一的框架，而下层则可以实现多种资源的控制。每个子系统的钩子函数如下:

```
1 struct cgroup_subsys {
2         struct cgroup_subsys_state *(*css_alloc)(struct cgroup *cgrp);
```

```
        int (*css_online)(struct cgroup *cgrp);
        void (*css_offline)(struct cgroup *cgrp);
        void (*css_free)(struct cgroup *cgrp);

        int (*can_attach)(struct cgroup *cgrp, struct cgroup_taskset *tset);
        void (*cancel_attach)(struct cgroup *cgrp, struct cgroup_taskset *tset);
        void (*attach)(struct cgroup *cgrp, struct cgroup_taskset *tset);
        void (*fork)(struct task_struct *task);
        void (*exit)(struct cgroup *cgrp, struct cgroup *old_cgrp,
                     struct task_struct *task);
        void (*bind)(struct cgroup *root);
        …… ……
}
```

3 cgroup文件系统

cgroup在Linux内核中是以文件系统的形式存在的，不过cgroup对应的这种文件系统与proc文件系统类似，都是只存在于内存中的“虚拟”文件系统。既然如此，就可以通过mount命令创建一个cgroup实例。

```
$ sudo mount -t cgroup -o memory memory_cgroup /dev/cgroup/
```

即在/dev/cgroup/下创建了一个memory子系统。接下来就可以通过：

```
$ cat /proc/filesystems | grep cgroup
```

可看到系统有了cgroup类型的文件系统。

当创建了一个cgroup实例后，对应的挂载点下会有一些文件，这些文件是用户与cgroup进行交互的接口。由于cgroup位于VFS层之下，因此用户可以通过统一的文件操作接口去读取或设置子系统的参数，当然也可以直接使用echo或者cat等命令。

参考:

1.Linux内核文档:

https://www.kernel.org/doc/Documentation/cgroups/cgroups.txt

- 上一篇: Linux下CPU的利用率

- 下一篇: Cgroup简介-子系统与层级

发表在 性能隔离
Tags: cgroup 性能隔离 虚拟化
若要跟踪这篇文章的任何更新，你可以使用 RSS 2.0 Feed. 你可以直接转到文章底部进行评论，Pinging目前已关闭

发表评论

姓名 (required)

电子邮件（不会被公开） (required)

站点

请回答问题: 3 + 9 = ?

发表评论

返回顶端
XHTML 1.0验证通过 | CSS3验证通过

GreenPark 2