

cgroups介绍及安装配置使用详解

1 cgroup简介

Cgroups是control groups的缩写，是Linux内核提供的一种可以限制、记录、隔离进程组（process groups）所使用的物理资源（如：cpu,memory,IO等等）的机制。最初由google的工程师提出，后来被整合进Linux内核。也是目前轻量级虚拟化技术 lxc（linux container）的基础之一。

2 cgroup作用

Cgroups最初的目标是为资源管理提供的一个统一的框架，既整合现有的cpuset等子系统，也为未来开发新的子系统提供接口。现在的cgroups适用于多种应用场景，从单个进程的资源控制，到实现操作系统层次的虚拟化（OS Level Virtualization）。Cgroups提供了以下功能：

- 1.限制进程组可以使用的资源数量（Resource limiting）。比如：memory子系统可以为进程组设定一个memory使用上限，一旦进程组使用的内存达到限额再申请内存，就会出发OOM（out of memory）。
- 2.进程组的优先级控制（Prioritization）。比如：可以使用cpu子系统为某个进程组分配特定cpu share。
- 3.记录进程组使用的资源数量（Accounting）。比如：可以使用cpuacct子系统记录某个进程组使用的cpu时间
- 4.进程组隔离（Isolation）。比如：使用ns子系统可以使不同的进程组使用不同的namespace，以达到隔离的目的，不同的进程组有各自的进程、网络、文件系统挂载空间。
- 5.进程组控制（Control）。比如：使用freezer子系统可以将进程组挂起和恢复。

3 cgroup相关概念

3.1 相关概念

- 1.任务（task）。在cgroups中，任务就是系统的一个进程。
- 2.控制族群（control group）。控制族群就是一组按照某种标准划分的进程。Cgroups中的资源控制都是以控制族群为单位实现。一个进程可以加入到某个控制族群，也从一个进程组迁移到另一个控制族群。一个进程组的进程可以使用cgroups以控制族群为单位分配的资源，同时受到cgroups以控制族群为单位设定的限制。
- 3.层级（hierarchy）。控制族群可以组织成hierarchical的形式，既一颗控制族群树。控制族群树上的子节点控制族群是父节点控制族群的孩子，继承父控制族群的特定的属性。
- 4.子系统（subsystem）。一个子系统就是一个资源控制器，比如cpu子系统就是控制cpu时间分配的一个控制器。子系统必须附加（attach）到一个层级上才能起作用，一个子系统附加到某个层级以后，这个层级上的所有控制族群都受到这个子系统的控制。

3.2 相互关系

1.每次在系统中创建新层级时，该系统中的所有任务都是那个层级的默认 cgroup（我们称之为 root cgroup，此cgroup在创建层级时自动创建，后面在该层级中创建的cgroup都是此cgroup的后代）的初始成员。

2.一个子系统最多只能附加到一个层级。

3.一个层级可以附加多个子系统

4.一个任务可以是多个cgroup的成员，但是这些cgroup必须不同的层级。

5.系统中的进程（任务）创建子进程（任务）时，该子任务自动成为其父进程所在 cgroup 的成员。然后可根据需要将该子任务移动到不同的 cgroup 中，但开始时它总是继承其父任务的cgroup。

4 cgroup子系统介绍

blkio -- 这个子系统为块设备设定输入/输出限制，比如物理设备（磁盘，固态硬盘，USB 等等）。

cpu -- 这个子系统使用调度程序提供对 CPU 的 cgroup 任务访问。

cpuacct -- 这个子系统自动生成 cgroup 中任务所使用的 CPU 报告。

cpuset -- 这个子系统为 cgroup 中的任务分配独立 CPU（在多核系统）和内存节点。

devices -- 这个子系统可允许或者拒绝 cgroup 中的任务访问设备。

freezer -- 这个子系统挂起或者恢复 cgroup 中的任务。

memory -- 这个子系统设定 cgroup 中任务使用的内存限制，并自动生成由那些任务使用的内存资源报告。

net_cls -- 这个子系统使用等级识别符（classid）标记网络数据包，可允许 Linux 流量控制程序（tc）识别从具体 cgroup 中生成的数据包。

ns -- 名称空间子系统。

5 cgroup安装 (centos下)

若系统未安装则进行安装，若已安装则进行更新。

```
[root@localhost ~]# yum install libcggroup
```

查看运行状态，并启动服务

```
[root@localhost ~]# service cgconfig status
```

Stopped

```
[root@localhost ~]# service cgconfig start
```

Starting cgconfig service: [OK]

```
[root@localhost ~]# service cgconfig status
```

Running

6 cgroup配置

6.1 配置文件介绍

6.1.1 cgroup配置文件所在位置

/etc/cgconfig.conf

6.1.2 默认配置文件内容

```
mount {  
    cpuset = /cgroup/cpuset;  
    cpu    = /cgroup/cpu;
```

```

    cpuacct = /cgroup/cpuacct;
    memory = /cgroup/memory;
    devices = /cgroup/devices;
    freezer = /cgroup/freezer;
    net_cls = /cgroup/net_cls;
    blkio = /cgroup/blkio;
}

```

相当于执行命令

```

mkdir /cgroup/cpuset
mount -t cgroup -o cpuset red /cgroup/cpuset
.....

```

```

mkdir /cgroup/blkio
mount -t cgroup -o cpuset red /cgroup/blkio

```

6.1.3 cgroup section的语法格式如下

```

group <name> {
    [<permissions>]
    <controller> {
        <param name> = <param value>;
        ...
    }
    ...}

```

name: 指定cgroup的名称

permissions: 可选项, 指定cgroup对应的挂载点文件系统的权限, root用户拥有所有权限。

controller: 子系统的名称

param name 和 param value: 子系统的属性及其属性值

7 cgroup实例分析 (限制mysql资源使用)

7.1 配置对mysql实例的资源限制

前提: mysql数据库已在机器上安装

7.1.1 修改cgconfig.conf文件

```

mount {
    cpuset    = /cgroup/cpuset;
    cpu       = /cgroup/cpu;
    cpuacct   = /cgroup/cpuacct;
    memory    = /cgroup/memory;
    blkio     = /cgroup/blkio;
}

group mysql_g1 {
    cpu {
        cpu.cfs_quota_us = 50000;
        cpu.cfs_period_us = 100000;
    }
}

```

```

cpuset {
    cpuset.cpus = "3";
    cpuset.mems = "0";
}
cpuacct{

}
memory {
    memory.limit_in_bytes=104857600;
    memory.swappiness=0;
    # memory.max_usage_in_bytes=104857600;
    # memory.oom_control=0;
}
blkio {
    blkio.throttle.read_bps_device="8:0 524288";
    blkio.throttle.write_bps_device="8:0 524288";
}
}

```

7.1.2 配置文件的部分解释。

cpu: cpu使用时间限额。

cpu.cfs_period_us和cpu.cfs_quota_us来限制该组中的所有进程在单位时间里可以使用的cpu时间。这里的cfs是完全公平调度器的缩写。cpu.cfs_period_us就是时间周期(微秒)，默认为100000，即百毫秒。cpu.cfs_quota_us就是在这期间内可使用的cpu时间(微秒)，默认-1，即无限制。(cfs_quota_us是cfs_period_us的两倍即可限定在双核上完全使用)。

cpuset: cpu绑定

我们限制该组只能在0一共1个超线程上运行。cpuset.mems是用来设置内存节点的。本例限制使用超线程0上的第四个cpu线程。

其实cgconfig也就是帮你把配置文件中的配置整理到/cgroup/cpuset这个目录里面，比如你需要动态设置mysql_group1/ cpuset.cpus的[CPU超线程](#)号，可以采用如下的办法。

```
[root@localhost ~]# echo "0" > mysql_group1/ cpuset.cpus
```

cpuacct: cpu资源报告

memory: 内存限制

内存限制我们主要限制了MySQL可以使用的内存最大大小memory.limit_in_bytes=256M。而设置swappiness为0是为了让操作系统不会将MySQL的内存匿名页交换出去。

blkio: BLOCK IO限额

```

blkio.throttle.read_bps_device="8:0 524288"; #每秒读数据上限
blkio.throttle.write_bps_device="8:0 524288"; #每秒写数据上限

```

其中8:0对应主设备号和副设备号，可以通过ls -l /dev/sda查看

```
[root@localhost /]# ls -l /dev/sda  
brw-rw----. 1 root disk 8, 0 Sep 15 04:19 /dev/sda
```

7.1.3 拓展知识

现在较新的服务器CPU都是numa结构<非一致内存访问结构 (NUMA: Non-Uniform Memory Access) >，使用numactl --hardware可以看到numa各个节点的CPU超线程号，以及对应的节点号。

本例结果如下：

```
[root@localhost /]# numactl --hardware  
available: 1 nodes (0)  
node 0 cpus: 0 1 2 3  
node 0 size: 1023 MB  
node 0 free: 68 MB  
node distances:  
node 0  
0: 10
```

以下是较高端服务器的numa信息，仅作参考。

```
[root@localhost ~]# numactl --hardware  
available: 4 nodes (0-3)  
node 0 cpus: 0 4 8 12 16 20 24 28 32 36 40 44 48 52 56 60  
node 0 size: 16338 MB  
node 0 free: 391 MB  
node 1 cpus: 1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61  
node 1 size: 16384 MB  
node 1 free: 133 MB  
node 2 cpus: 2 6 10 14 18 22 26 30 34 38 42 46 50 54 58 62  
node 2 size: 16384 MB  
node 2 free: 137 MB  
node 3 cpus: 3 7 11 15 19 23 27 31 35 39 43 47 51 55 59 63  
node 3 size: 16384 MB  
node 3 free: 186 MB  
node distances:  
node 0 1 2 3  
0: 10 20 30 20  
1: 20 10 20 30  
2: 30 20 10 20  
3: 20 30 20 10
```

7.1.4 修改cgrules.conf文件

```
[root@localhost ~]# vi /etc/cgrules.conf  
# /etc/cgrules.conf  
#The format of this file is described in cgrules.conf(5)  
#manual page.  
#
```

Example:

```
#<user>      <controllers> <destination>
#@student    cpu,memory   usergroup/student/
#peter       cpu         test1/
#%           memory      test2/
*:/usr/local/mysql/bin/mysqld * mysql_g1
```

注：共分为3个部分，分别为需要限制的实例，限制的内容（如cpu，memory），挂载目标。

7.2 使配置生效

```
[root@localhost ~]# /etc/init.d/cgconfig restart
```

```
Stopping cgconfig service: [ OK ]
```

```
Starting cgconfig service: [ OK ]
```

```
[root@localhost ~]# /etc/init.d/cgred restart
```

```
Stopping CGroup Rules Engine Daemon... [ OK ]
```

```
Starting CGroup Rules Engine Daemon: [ OK ]
```

注：重启顺序为cgconfig -> cgred，更改配置文件后两个服务需要重启，且顺序不能错。

7.3 启动MySQL，查看MySQL是否处于cgroup的限制中

```
[root@localhost ~]# ps -eo pid,cgroup,cmd | grep -i mysqld
```

```
29871 blkio:/;net_cls:/;freezer:/;devices:/;memory:/;cpuacct:/;cpu:/;cpuset:/ /bin/sh
./bin/mysqld_safe --defaults-file=/etc/my.cnf --basedir=/usr/local/mysql/ --
datadir=/usr/local/mysql/data/
30219 blkio:/;net_cls:/;freezer:/;devices:/;memory:/;cpuacct:/;cpu:/;cpuset:/mysql_g1
/usr/local/mysql/bin/mysqld --defaults-file=/etc/my.cnf --basedir=/usr/local/mysql/ --
datadir=/usr/local/mysql/data/ --plugin-dir=/usr/local/mysql/lib/plugin --user=mysql --log-
error=/usr/local/mysql/data/localhost.localdomain.err --pid-
file=/usr/local/mysql/data/localhost.localdomain.pid --socket=/tmp/mysql.sock --port=3306
30311 blkio:/;net_cls:/;freezer:/;devices:/;memory:/;cpuacct:/;cpu:/;cpuset:/ grep -i mysqld
```

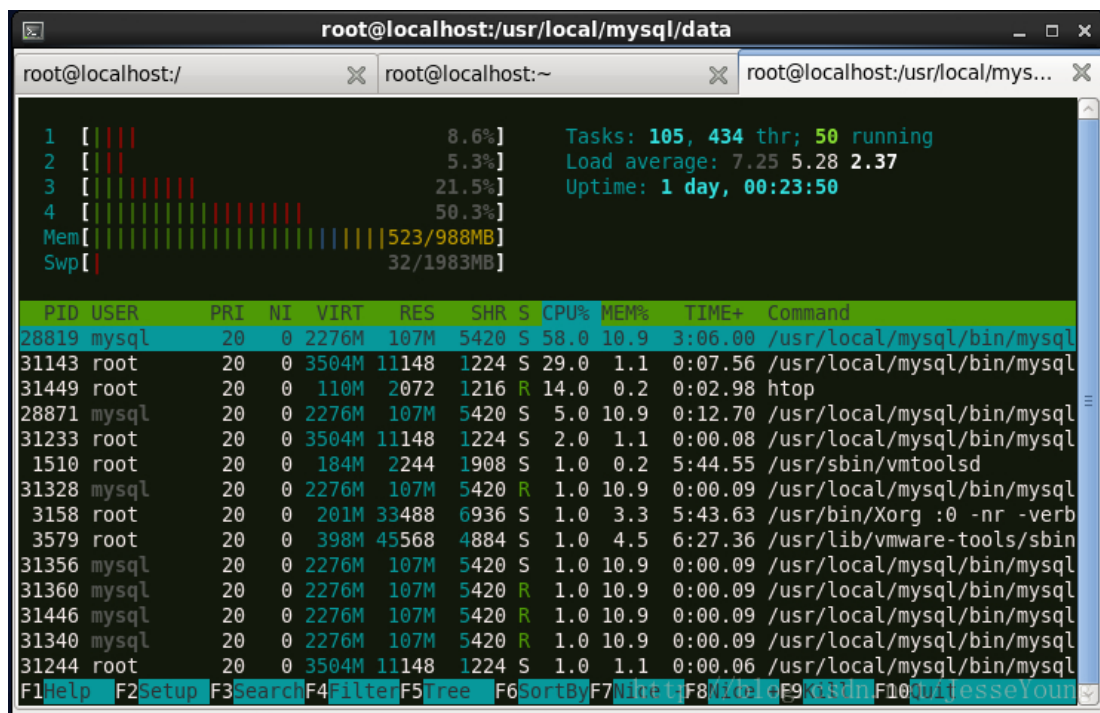
7.4 资源限制验证

使用mysqlslap对mysql进行压力测试，看mysql使用资源是否超过限制。

7.4.1 在shell窗口1用mysqlslap对mysql进行压力测试。

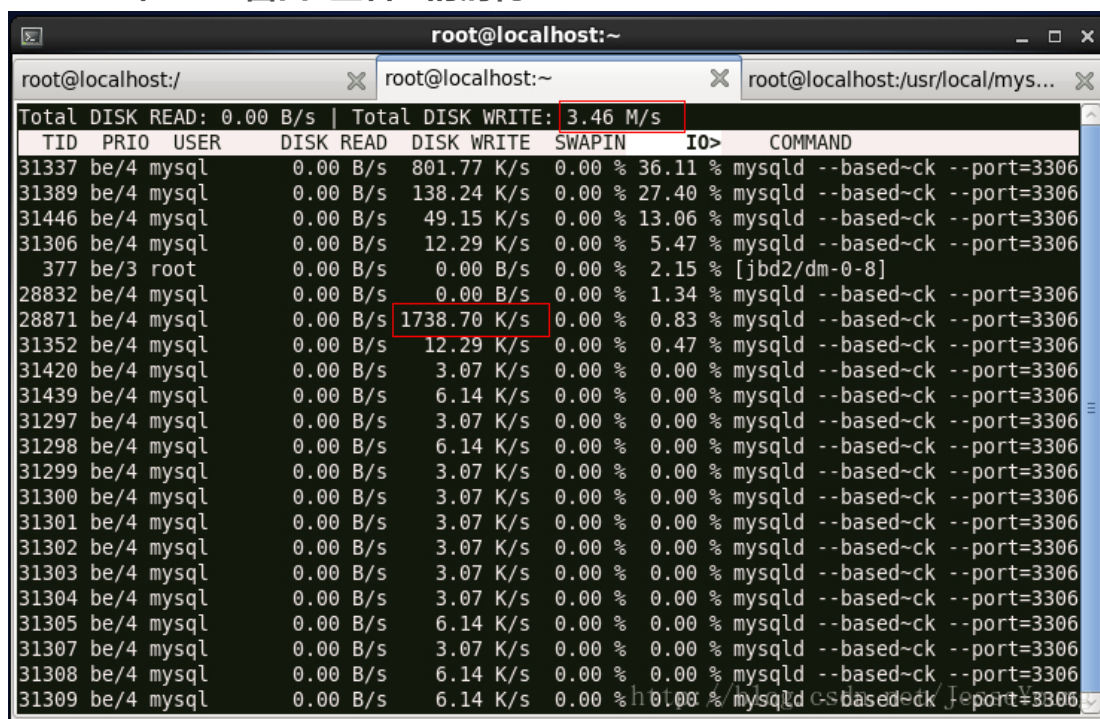
```
[root@localhost /]# /usr/local/mysql/bin/mysqlslap --defaults-file=/etc/my.cnf --
concurrency=150 --iterations=1 --number-int-cols=8 --auto-generate-sql --auto-generate-
sql-load-type=mixed --engine=innodb --number-of-queries=100000 -ujesse -pjesse --
number-char-cols=35 --auto-generate-sql-add-autoincrement --debug-info -P3306 -
h127.0.0.1
```

7.4.2 在shell窗口2查看mysql对cpu，内存的使用



可见：cpu限制在了第四个核心上，且对第四个核心的使用限制在50%。

7.4.3 在shell窗口3查看io的消耗



可见：mysql对io的读及写消耗均限制在2M每秒以内。

8 cgroup实例分析（手工动态验证）

还原配置文件/etc/cgconfig.conf及/etc/cgrules.conf 为默认配置。测试实例依然为mysql，测试工具为mysqlslap。

开启cgconfig及cgrules 服务。

[root@localhost /]# /etc/init.d/cgconfig restart

Stopping cgconfig service:

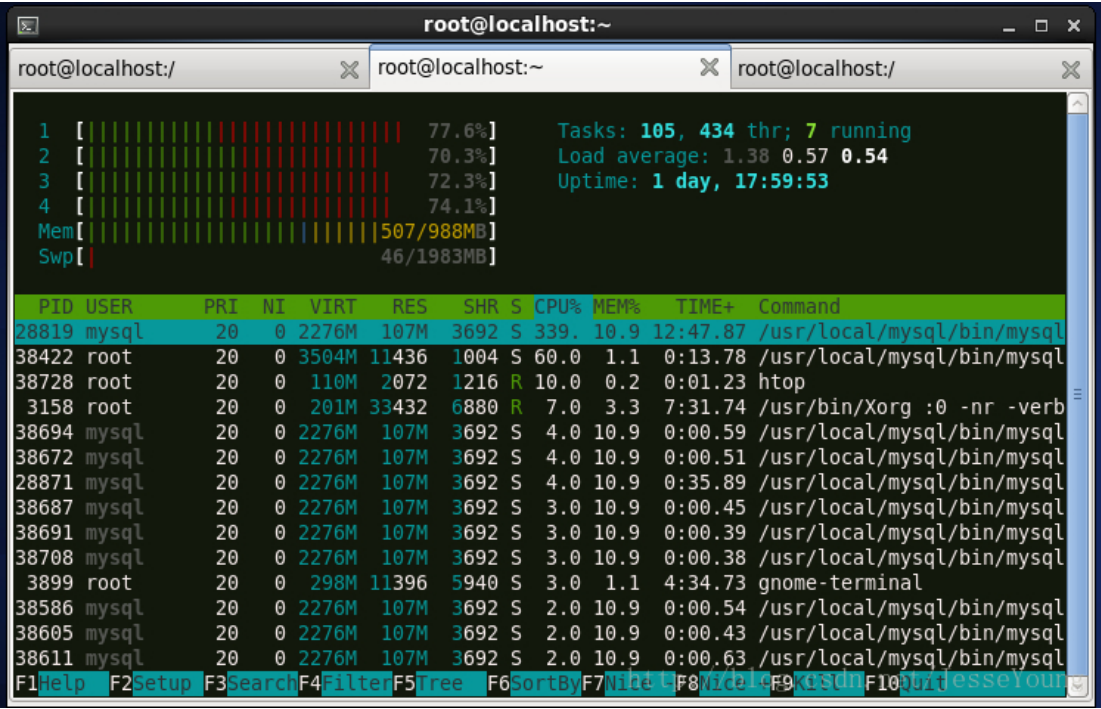
[OK]

Starting cgconfig service: [OK]
[root@localhost /]# /etc/init.d/cgred restart
Stopping CGroup Rules Engine Daemon... [OK]
Starting CGroup Rules Engine Daemon: [OK]

开启mysqlslap压力测试程序。

[root@localhost /]# /usr/local/mysql/bin/mysqlslap --defaults-file=/etc/my.cnf --
concurrency=150 --iterations=1 --number-int-cols=8 --auto-generate-sql --auto-generate-
sql-load-type=mixed --engine=innodb --number-of-queries=100000 -ujesse -pjesse --
number-char-cols=35 --auto-generate-sql-add-autoincrement --debug-info -P3306 -
h127.0.0.1

通过htop查看资源消耗。

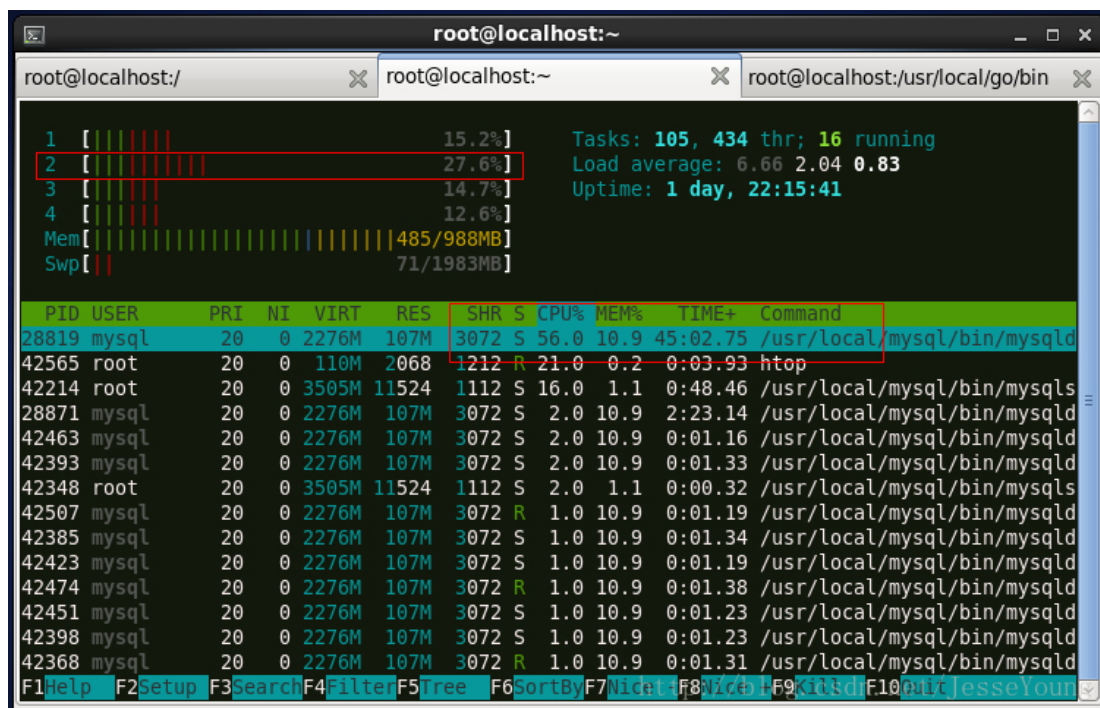


8.1 cpu限制实例

限制mysql使用一个核，如第2个核，且对该核的使用不超过50%

[root@localhost /]# mkdir -p /cgroup/cpu/foo/
[root@localhost /]# mkdir -p /cgroup/cpuset/foo/
[root@localhost /]# echo 50000 > /cgroup/cpu/foo/cpu.cfs_quota_us
[root@localhost /]# echo 100000 > /cgroup/cpu/foo/cpu.cfs_period_us
[root@localhost /]# echo "0" > /cgroup/cpuset/foo/cpuset.mems
[root@localhost /]# echo "1" > /cgroup/cpuset/foo/cpuset.cpus
[root@localhost /]# echo 28819 > /cgroup/cpu/foo/tasks

其中：28819为mysqld的进程号。



8.2 内存限制实例

限制mysql使用内存为不超过512M

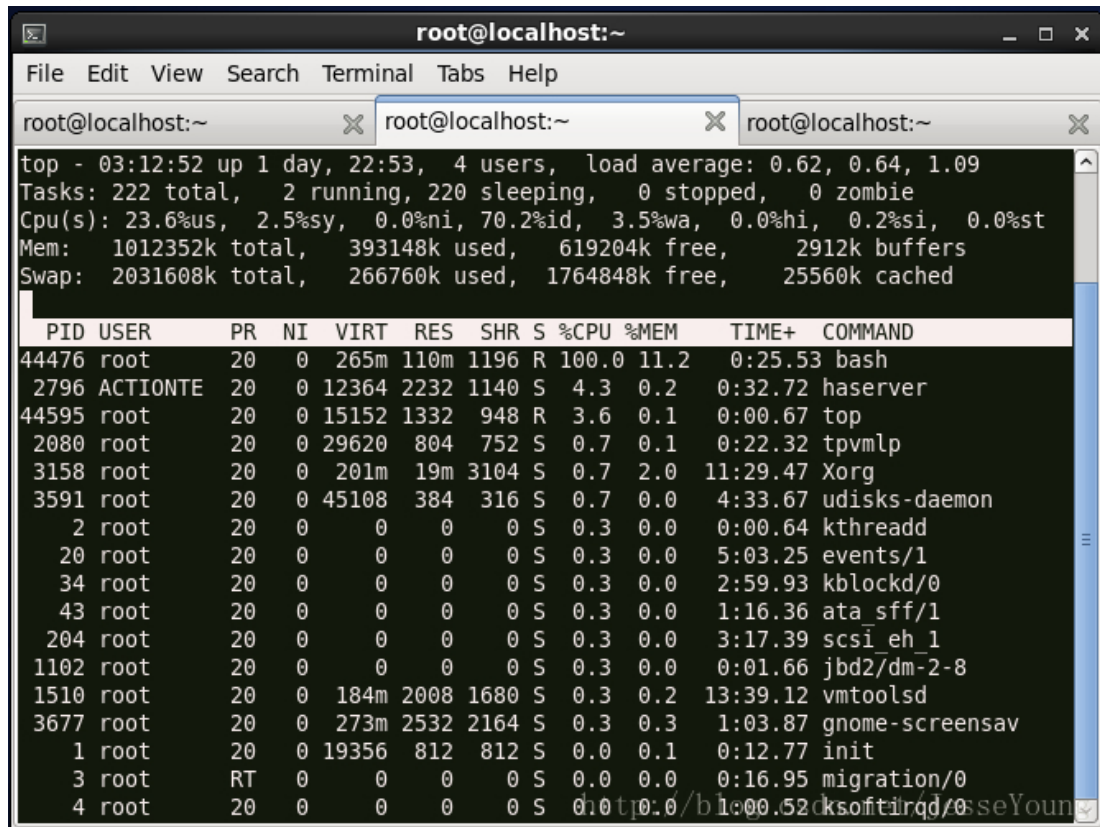
跑一个消耗内存脚本

```
x='a'
```

```
while [ True ];do
```

```
  x=$xx
```

```
done;
```

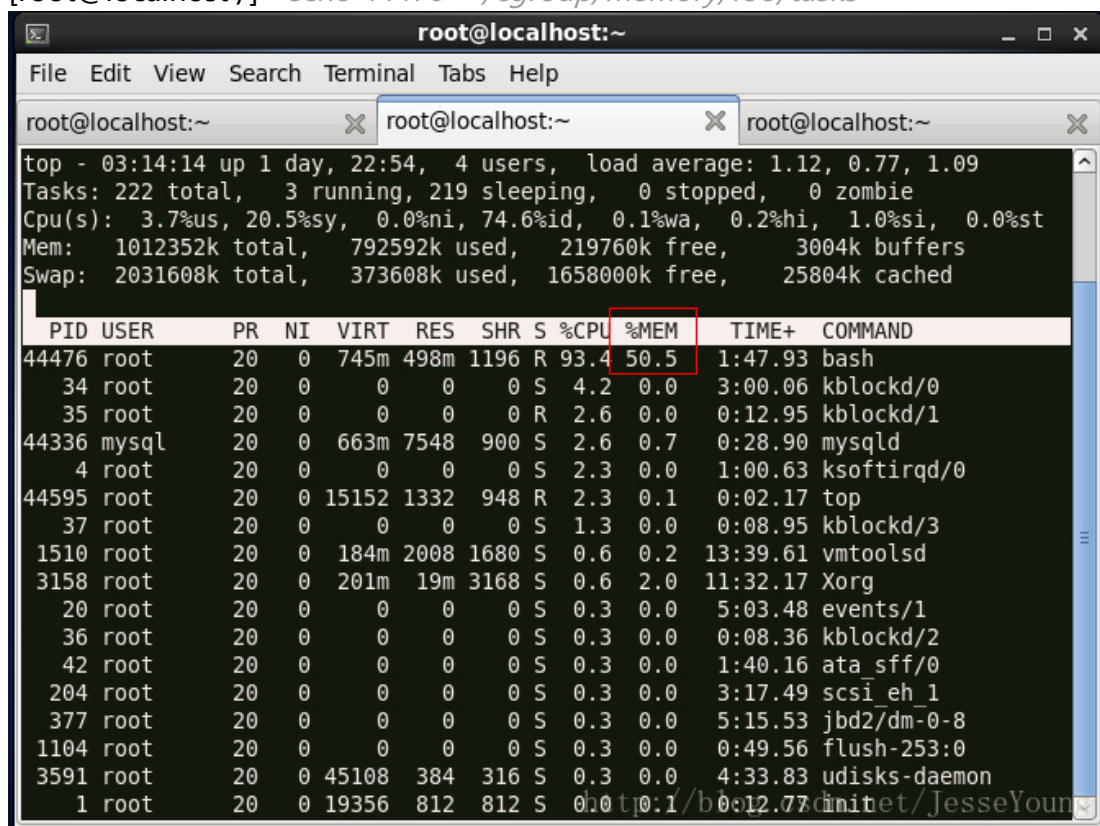


内存的消耗在不断增加，对其进行限制，使其使用内存在500M以内

```
[root@localhost ~]# mkdir -p /cgroup/memory/foo
```

```
[root@localhost ~]# echo 524288000 > /cgroup/memory/foo/memory.limit_in_bytes
```

```
[root@localhost ~]# echo 44476 > /cgroup/memory/foo/tasks
```



```
top - 03:14:14 up 1 day, 22:54, 4 users, load average: 1.12, 0.77, 1.09
Tasks: 222 total, 3 running, 219 sleeping, 0 stopped, 0 zombie
Cpu(s): 3.7%us, 20.5%sy, 0.0%ni, 74.6%id, 0.1%wa, 0.2%hi, 1.0%si, 0.0%st
Mem: 1012352k total, 792592k used, 219760k free, 3004k buffers
Swap: 2031608k total, 373608k used, 1658000k free, 25804k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
44476	root	20	0	745m	498m	1196	R	93.4	50.5	1:47.93	bash
34	root	20	0	0	0	0	S	4.2	0.0	3:00.06	kblockd/0
35	root	20	0	0	0	0	R	2.6	0.0	0:12.95	kblockd/1
44336	mysql	20	0	663m	7548	900	S	2.6	0.7	0:28.90	mysqld
4	root	20	0	0	0	0	S	2.3	0.0	1:00.63	ksoftirqd/0
44595	root	20	0	15152	1332	948	R	2.3	0.1	0:02.17	top
37	root	20	0	0	0	0	S	1.3	0.0	0:08.95	kblockd/3
1510	root	20	0	184m	2008	1680	S	0.6	0.2	13:39.61	vmtoolsd
3158	root	20	0	201m	19m	3168	S	0.6	2.0	11:32.17	Xorg
20	root	20	0	0	0	0	S	0.3	0.0	5:03.48	events/1
36	root	20	0	0	0	0	S	0.3	0.0	0:08.36	kblockd/2
42	root	20	0	0	0	0	S	0.3	0.0	1:40.16	ata_sff/0
204	root	20	0	0	0	0	S	0.3	0.0	3:17.49	scsi_eh_1
377	root	20	0	0	0	0	S	0.3	0.0	5:15.53	jbd2/dm-0-8
1104	root	20	0	0	0	0	S	0.3	0.0	0:49.56	flush-253:0
3591	root	20	0	45108	384	316	S	0.3	0.0	4:33.83	udisks-daemon
1	root	20	0	19356	812	812	S	0.0	0.0	0:00.00	init

内存使用得到了有效控制。

8.3 IO限制实例

跑一个消耗IO的测试

```
[root@localhost ~]# dd if=/dev/sda of=/dev/null
```

通过iotop看io占用情况，磁盘读取速度到了50M/s

Total DISK READ: 51.31 M/s Total DISK WRITE: 0.00 B/s							
TID	PRI	USER	DISK READ	DISK WRITE	SWAPIN	IO>	COMMAND
45015	be/4	root	51.22 M/s	0.00 B/s	0.00 %	1.14 %	dd if=/dev/...
44350	be/4	mysql	93.67 K/s	0.00 B/s	0.00 %	0.27 %	mysqld ---port=3306
1	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	init
2	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[kthreadd]
3	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/0]
4	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[ksoftirqd/0]
5	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/0]
6	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[watchdog/0]
7	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/1]
8	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/1]
9	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[ksoftirqd/1]
10	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[watchdog/1]
11	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/2]
12	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/2]
13	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[ksoftirqd/2]
14	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[watchdog/2]
15	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/3]
16	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[migration/3]
17	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[ksoftirqd/3]
18	rt/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[watchdog/3]
19	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[events/0]
20	be/4	root	0.00 B/s	0.00 B/s	0.00 %	0.00 %	[events/1]

限制读取速度为10M/S

```
[root@localhost ~]# mkdir -p /cgroup/blkio/foo
```

```
[root@localhost ~]# echo '8:0 10485760' > /cgroup/blkio/foo/blkio.throttle.read_bps_device
```

```
[root@localhost ~]# echo 45033 > /cgroup/blkio/foo/tasks
```

注1:45033为dd的进程号

注2: 8:0对应主设备号和副设备号，可以通过ls -l /dev/sda查看

```
[root@localhost ~]# ls -l /dev/sda
```

```
brw-rw----. 1 root disk 8, 0 Sep 15 04:19 /dev/sda
```

9 cgroup小结

使用cgroup临时对进程进行调整，直接通过命令即可，如果要持久化对进程进行控制，即重启后依然有效，需要写进配置文件/etc/cgconfig.conf及/etc/cgrules.conf

原文地址: <http://blog.csdn.net/jesseyoung/article/details/39077829>

博客主页: <http://blog.csdn.net/jesseyoung>
