该文件中的所有值都是从系统启动开始累计到当前时刻。该文件只给出了所有CPU的集合信息，不能该出每个CPU的信息。  
  
[root@localhost ~]# cat /proc/loadavg   
4.61 4.36 4.15 9/84 5662  
  
每个值的含义为：  
参数 解释  
lavg\_1 (4.61) 1-分钟平均负载  
lavg\_5 (4.36) 5-分钟平均负载  
lavg\_15(4.15) 15-分钟平均负载  
nr\_running (9) 在采样时刻，运行队列的任务的数目，与/proc/stat的procs\_running表示相同意思  
nr\_threads (84) 在采样时刻，系统中活跃的任务的个数（不包括运行已经结束的任务）  
last\_pid(5662) 最大的pid值，包括轻量级进程，即线程。  
  
假设当前有两个CPU，则每个CPU的当前任务数为4.61/2=2.31 

[**PROC系列之---/proc/stat/**](http://blog.csdn.net/zjl_1026_2001/archive/2008/04/15/2294057.aspx)

包含了所有CPU活动的信息，该文件中的所有值都是从系统启动开始累计到当前时刻。  
  
[work@builder ~]$ cat /proc/stat  
cpu 432661 13295 86656 422145968 171474 233 5346  
cpu0 123075 2462 23494 105543694 16586 0 4615  
cpu1 111917 4124 23858 105503820 69697 123 371  
cpu2 103164 3554 21530 105521167 64032 106 334  
cpu3 94504 3153 17772 105577285 21158 4 24  
intr 1065711094 1057275779 92 0 6 6 0 4 0 3527 0 0 0 70 0 20 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 7376958 0 0 0 0 0 0 0 1054602 0 0 0 0 0 0 0 30 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0  
ctxt 19067887  
btime 1139187531  
processes 270014  
procs\_running 1  
procs\_blocked 0  
  
  
输出解释  
CPU 以及CPU0、CPU1、CPU2、CPU3每行的每个参数意思（以第一行为例）为：  
参数 解释  
user (432661) 从系统启动开始累计到当前时刻，用户态的CPU时间（单位：jiffies） ，不包含 nice值为负进程。1jiffies=0.01秒  
nice (13295) 从系统启动开始累计到当前时刻，nice值为负的进程所占用的CPU时间（单位：jiffies）   
system (86656) 从系统启动开始累计到当前时刻，核心时间（单位：jiffies）   
idle (422145968) 从系统启动开始累计到当前时刻，除硬盘IO等待时间以外其它等待时间（单位：jiffies）   
iowait (171474) 从系统启动开始累计到当前时刻，硬盘IO等待时间（单位：jiffies） ，  
irq (233) 从系统启动开始累计到当前时刻，硬中断时间（单位：jiffies）   
softirq (5346) 从系统启动开始累计到当前时刻，软中断时间（单位：jiffies）   
  
CPU时间=user+system+nice+idle+iowait+irq+softirq  
  
“intr”这行给出中断的信息，第一个为自系统启动以来，发生的所有的中断的次数；然后每个数对应一个特定的中断自系统启动以来所发生的次数。  
“ctxt”给出了自系统启动以来CPU发生的上下文交换的次数。  
“btime”给出了从系统启动到现在为止的时间，单位为秒。  
“processes (total\_forks) 自系统启动以来所创建的任务的个数目。  
“procs\_running”：当前运行队列的任务的数目。  
“procs\_blocked”：当前被阻塞的任务的数目。 

[**PROC系列之---/proc/pid/stat**](http://blog.csdn.net/zjl_1026_2001/archive/2008/04/15/2294067.aspx)

/proc/ /stat  
包含了所有CPU活跃的信息，该文件中的所有值都是从系统启动开始累计到当前时刻。  
  
[root@localhost ~]# cat /proc/6873/stat  
6873 (a.out) R 6723 6873 6723 34819 6873 8388608 77 0 0 0 41958 31 0 0 25 0 3 0 5882654 1409024 56 4294967295 134512640 134513720 3215579040 0 2097798 0 0 0 0 0 0 0 17 0 0 0 [root@localhost ~]#  
  
  
每个参数意思为：  
参数 解释  
pid=6873 进程(包括轻量级进程，即线程)号  
comm=a.out 应用程序或命令的名字  
task\_state=R 任务的状态，R:runnign, S:sleeping (TASK\_INTERRUPTIBLE), D:disk sleep (TASK\_UNINTERRUPTIBLE), T: stopped, T:tracing stop,Z:zombie, X:dead  
ppid=6723 父进程ID  
pgid=6873 线程组号  
sid=6723 c该任务所在的会话组ID  
tty\_nr=34819(pts/3) 该任务的tty终端的设备号，INT（34817/256）=主设备号，（34817-主设备号）=次设备号  
tty\_pgrp=6873 终端的进程组号，当前运行在该任务所在终端的前台任务(包括shell 应用程序)的PID。  
task->flags=8388608 进程标志位，查看该任务的特性  
min\_flt=77 该任务不需要从硬盘拷数据而发生的缺页（次缺页）的次数  
cmin\_flt=0 累计的该任务的所有的waited-for进程曾经发生的次缺页的次数目  
maj\_flt=0 该任务需要从硬盘拷数据而发生的缺页（主缺页）的次数  
cmaj\_flt=0 累计的该任务的所有的waited-for进程曾经发生的主缺页的次数目  
utime=1587 该任务在用户态运行的时间，单位为jiffies  
stime=1 该任务在核心态运行的时间，单位为jiffies  
cutime=0 累计的该任务的所有的waited-for进程曾经在用户态运行的时间，单位为jiffies  
cstime=0 累计的该任务的所有的waited-for进程曾经在核心态运行的时间，单位为jiffies  
priority=25 任务的动态优先级  
nice=0 任务的静态优先级  
num\_threads=3 该任务所在的线程组里线程的个数  
it\_real\_value=0 由于计时间隔导致的下一个 SIGALRM 发送进程的时延，以 jiffy 为单位.  
start\_time=5882654 该任务启动的时间，单位为jiffies  
vsize=1409024（page） 该任务的虚拟地址空间大小  
rss=56(page) 该任务当前驻留物理地址空间的大小  
Number of pages the process has in real memory,minu 3 for administrative purpose.  
这些页可能用于代码，数据和栈。  
rlim=4294967295（bytes） 该任务能驻留物理地址空间的最大值  
start\_code=134512640 该任务在虚拟地址空间的代码段的起始地址  
end\_code=134513720 该任务在虚拟地址空间的代码段的结束地址  
start\_stack=3215579040 该任务在虚拟地址空间的栈的结束地址  
kstkesp=0 esp(32 位堆栈指针) 的当前值, 与在进程的内核堆栈页得到的一致.  
kstkeip=2097798 指向将要执行的指令的指针, EIP(32 位指令指针)的当前值.  
pendingsig=0 待处理信号的位图，记录发送给进程的普通信号  
block\_sig=0 阻塞信号的位图  
sigign=0 忽略的信号的位图  
sigcatch=082985 被俘获的信号的位图  
wchan=0 如果该进程是睡眠状态，该值给出调度的调用点  
nswap 被swapped的页数，当前没用  
cnswap 所有子进程被swapped的页数的和，当前没用  
exit\_signal=17 该进程结束时，向父进程所发送的信号  
task\_cpu(task)=0 运行在哪个CPU上  
task\_rt\_priority=0 实时进程的相对优先级别  
task\_policy=0 进程的调度策略，0=非实时进程，1=FIFO实时进程；2=RR实时进程 

[**PROC系列之---/proc/pid/statm**](http://blog.csdn.net/zjl_1026_2001/archive/2008/04/15/2294072.aspx)

/proc/ /statm  
包含了所有CPU活跃的信息，该文件中的所有值都是从系统启动开始累计到当前时刻。  
  
[root@localhost ~]# cat /proc/self/statm  
654 57 44 0 0 334 0  
  
  
输出解释  
CPU 以及CPU0。。。的每行的每个参数意思（以第一行为例）为：  
参数 解释 /proc/ /status  
Size (pages) 任务虚拟地址空间的大小 VmSize/4  
Resident(pages) 应用程序正在使用的物理内存的大小 VmRSS/4  
Shared(pages) 共享页数 0  
Trs(pages) 程序所拥有的可执行虚拟内存的大小 VmExe/4  
Lrs(pages) 被映像到任务的虚拟内存空间的库的大小 VmLib/4  
Drs(pages) 程序数据段和用户态的栈的大小 （VmData+ VmStk ）4  
dt(pages) 0 

[**PROC系列之---/proc/pid/status**](http://blog.csdn.net/zjl_1026_2001/archive/2008/04/15/2294036.aspx)

/proc/ /status  
包含了所有CPU活跃的信息，该文件中的所有值都是从系统启动开始累计到当前时刻。  
  
[root@localhost ~]# cat /proc/self/status  
Name: cat  
State: R (running)  
SleepAVG: 88%  
Tgid: 5783  
Pid: 5783  
PPid: 5742  
TracerPid: 0  
Uid: 0 0 0 0  
Gid: 0 0 0 0  
FDSize: 256  
Groups: 0 1 2 3 4 6 10   
VmSize: 6588 kB  
VmLck: 0 kB  
VmRSS: 400 kB  
VmData: 144 kB  
VmStk: 2040 kB  
VmExe: 14 kB  
VmLib: 1250 kB  
StaBrk: 0804e000 kB  
Brk: 088df000 kB  
StaStk: bfe03270 kB  
ExecLim: 0804c000  
Threads: 1  
SigPnd: 0000000000000000  
ShdPnd: 0000000000000000  
SigBlk: 0000000000000000  
SigIgn: 0000000000000000  
SigCgt: 0000000000000000  
CapInh: 0000000000000000  
CapPrm: 00000000fffffeff  
CapEff: 00000000fffffeff  
  
  
输出解释  
参数 解释  
Name 应用程序或命令的名字  
State 任务的状态，运行/睡眠/僵死/  
SleepAVG 任务的平均等待时间(以nanosecond为单位)，交互式任务因为休眠次数多、时间长，它们的 sleep\_avg 也会相应地更大一些，所以计算出来的优先级也会相应高一些。  
Tgid 线程组号  
Pid 任务ID  
Ppid 父进程ID  
TracerPid 接收跟踪该进程信息的进程的ID号  
Uid Uid euid suid fsuid  
Gid Gid egid sgid fsgid  
FDSize 文件描述符的最大个数，file->fds  
Groups   
VmSize(KB) 任务虚拟地址空间的大小 (total\_vm-reserved\_vm)，其中total\_vm为进程的地址空间的大小，reserved\_vm：进程在预留或特殊的内存间的物理页  
VmLck(KB) 任务已经锁住的物理内存的大小。锁住的物理内存不能交换到硬盘 (locked\_vm)  
VmRSS(KB) 应用程序正在使用的物理内存的大小，就是用ps命令的参数rss的值 (rss)  
VmData(KB) 程序数据段的大小（所占虚拟内存的大小），存放初始化了的数据； (total\_vm-shared\_vm-stack\_vm)  
VmStk(KB) 任务在用户态的栈的大小 (stack\_vm)  
VmExe(KB) 程序所拥有的可执行虚拟内存的大小，代码段，不包括任务使用的库 (end\_code-start\_code)  
VmLib(KB) 被映像到任务的虚拟内存空间的库的大小 (exec\_lib)  
VmPTE 该进程的所有页表的大小，单位：kb  
Threads 共享使用该信号描述符的任务的个数，在POSIX多线程序应用程序中，线程组中的所有线程使用同一个信号描述符。  
SigQ 待处理信号的个数  
SigPnd 屏蔽位，存储了该线程的待处理信号  
ShdPnd 屏蔽位，存储了该线程组的待处理信号  
SigBlk 存放被阻塞的信号  
SigIgn 存放被忽略的信号  
SigCgt 存放被俘获到的信号  
CapInh Inheritable，能被当前进程执行的程序的继承的能力  
CapPrm Permitted，进程能够使用的能力，可以包含CapEff中没有的能力，这些能力是被进程自己临时放弃的，CapEff是CapPrm的一个子集，进程放弃没有必要的能力有利于提高安全性  
CapEff Effective，进程的有效能力  
  
  
范例 1  
可 以看出该应用程序的正文段(1KB)很小，说明代码很少，是依靠库（1251KB）来执行。栈（138KB）适中，说明没有太多许多嵌套函数或特别多的临 时变量。VmLck为0说明进程没有锁住任何页。VmRSS表示当前进程使用的物理内存为2956KB。当进程开始使用已经申请的但还没有用的内存时， VmRSS的值开始增大，但是VmSize保持不变。  
[root@localhost 1]# cat /proc/4668/status  
Name: gam\_server  
State: S (sleeping)  
SleepAVG: 88%  
Tgid: 31999  
Pid: 31999  
PPid: 1  
TracerPid: 0  
Uid: 0 0 0 0  
Gid: 0 0 0 0  
FDSize: 256  
Groups: 0 1 2 3 4 6 10  
VmSize: 2136 kB  
VmLck: 0 kB  
VmRSS: 920 kB  
VmData: 148 kB  
VmStk: 88 kB  
VmExe: 44 kB  
VmLib: 1820 kB  
VmPTE: 20 kB  
Threads: 1  
SigQ: 1/2047  
SigPnd: 0000000000000000  
ShdPnd: 0000000000000000  
SigBlk: 0000000000000000  
SigIgn: 0000000000001006  
SigCgt: 0000000210000800  
CapInh: 0000000000000000  
CapPrm: 00000000fffffeff  
CapEff: 00000000fffffeff  
[root@localhost 31999]#