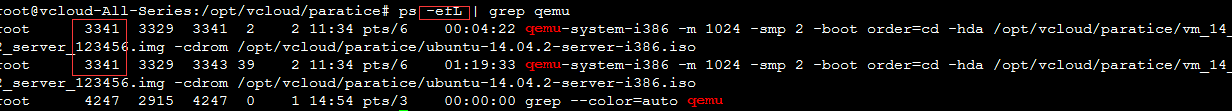
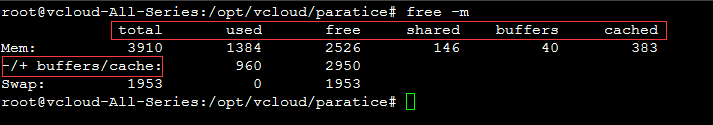
一个虚拟机相当于一个进程、分配给虚拟机的cpu个数相当于进程中的线程的个数。

查看qemu模拟的虚拟机cpu的个数 vcpu



C:\Users\huzhengchuan\AppData\Roaming\Tencent\Users\875161027\QQ\WinTemp\RichOle\Q534_RARXTBWV(1P))QUL8H.png

附录：



Mem：表示物理内存统计

total : OS所有可用内存大小

used : 已经使用的内存大小

free： 空闲的内存大小

shared 多个进程共享的内存总额

buffers Buffer Cache系统分配但未被使用的buffers 数量。

cached Page Cache系统分配但未被使用的cache 数量。

-buffers/cache 的内存数:实际使用的buffers 与cache 总量，也是实际使用的内存总量。used - buffers – cached

+buffers/cache 的内存数:未被使用的buffers 与cache 和未被分配的内存之和，这就是系统当前实际可用内存。free + buffers + cached

释放内存.

1. 手动执行sync命令(描述:sync 命令运行 sync 子例程。如果必须停止系统，则运行 sync 命令以确保文件系统的完整性。sync 命令将所有未写的系统缓冲区写到磁盘中，包含已修改的 i-node、已延迟的块 I/O 和读写映射文件
2. echo 3 >/proc/sys/vm/drop\_caches

* buffer 与cache 的区别

　　A buffer is something that has yet to be "written" to disk. A cache is something that has been "read" from the disk and stored for later use.

* 为了提高磁盘存取效率, Linux做了一些精心的设计, 除了对dentry进行缓存(用于VFS,加速文件路径名到inode的转换), 还采取了两种主要Cache方式：Buffer Cache和Page Cache。前者针对磁盘块的读写，后者针对文件inode的读写。这些Cache有效缩短了 I/O系统调用(比如read,write,getdents)的时间。"

/proc/sys/vm/drop\_caches (since Linux 2.6.16) Writing to this file causes the kernel to drop clean caches,dentries and inodes from memory, causing that memory to become free.To free pagecache, use echo 1 >/proc/sys/vm/drop\_caches; to free dentries and inodes, use echo 2 >/proc/sys/vm/drop\_caches; to free pagecache, dentries and inodes, use echo 3 > /proc/sys/vm/drop\_caches. Because this is a non-destructive operation and dirty objects

附录

首先了解一下需要用到的命令：

查看CPU的基本信息：

查看物理CPU的个数   
#cat /proc/cpuinfo |grep "physical id"|sort |uniq|wc -l   
查看逻辑CPU的个数   
#cat /proc/cpuinfo |grep "processor"|wc -l   
查看CPU是几核   
#cat /proc/cpuinfo |grep "cores"|uniq   
查看CPU的主频   
#cat /proc/cpuinfo |grep MHz|uniq

指定进程到具体CPU上运行：

# taskset

-p，  设定一个已存在的pid，而不是重新开启的一个新任务

-c，  指定一个处理，可以指定多个，以逗号分隔，也可指定方位，如2,4,5,6-8

1，切换某个进程到指定的CPU上

taskset -cp 3 13290

2，让某程序运行在指定的CPU上

taskset -c 1,2,4-7 tar jcf test.tar.gz test

查看物理主机的CPU个数为16个：

|  |  |
| --- | --- |
| 1  2 | [root@test8 home]# cat /proc/cpuinfo |grep "processor"|wc -l  16 |

将CPU14和CPU15进行隔离：修改/boot/grub/grub.conf文件，在kernel一行的后面添加**isolcpus=14,15**

|  |  |
| --- | --- |
| 1  2  3 | root (hd0,0)  kernel /boot/vmlinuz-2.6.32-279.2.1.el6.x86\_64 ro root=UUID=4f8c5d8b-fc5f-49d2-96c8-71effe86e58f rd\_NO\_LUKS rd\_NO\_LVM rd\_NO\_MD rd\_NO\_DM LANG=zh\_CN.UTF-8 KEYBOARDTYPE=pc KEYTABLE=us crashkernel=auto rhgb quiet isolcpus=14,15  initrd /boot/initramfs-2.6.32-279.2.1.el6.x86\_64.img |

注意：grub.conf是启动的配置文件，非常重要，提示在修改前对grub.conf做一下备份

配置grub.conf完成后，重新启动主机，使用命令ps -eLo psr | grep 0 | wc -l可以查看CPU的使用情况

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | [root@test8 home]# ps -eLo psr | grep 0 | wc -l  131  [root@test8 home]# ps -eLo psr | grep 1 | wc -l  310  [root@test8 home]# ps -eLo psr | grep 2 | wc -l  84  [root@test8 home]# ps -eLo psr | grep 14 | wc -l  21  [root@test8 home]# ps -eLo psr | grep 15 | wc -l  21 |

显示CPU14和CPU15运行了21个线程（这些线程都是系统启动的守护线程，CPU14和CPU15中运行的相同）

创建两个相同的虚拟机，查看系统中的虚拟机程序运行的进程号

|  |  |
| --- | --- |
| 1 | [root@test8 home]# ps -eLo ruser,pid,ppid,lwp,psr,args | grep qemu | grep -v grep |
| 1  2  3  4  5  6 | qemu     11535     1 11535   3 /usr/libexec/qemu-kvm -name centosb -S -M rhel6.4.0 -enable-kvm -m 1024 -smp 1,sockets=1,cores=1,threads=1 ......  qemu     11535     1 11556   0 /usr/libexec/qemu-kvm -name centosb -S -M rhel6.4.0 -enable-kvm -m 1024 -smp 1,sockets=1,cores=1,threads=1 ......  qemu     11535     1 13001   3 /usr/libexec/qemu-kvm -name centosb -S -M rhel6.4.0 -enable-kvm -m 1024 -smp 1,sockets=1,cores=1,threads=1 ......  qemu     13224     1 13224  15 /usr/libexec/qemu-kvm -name test1 -S -M rhel6.4.0 -enable-kvm -m 1024 -smp 1,sockets=1,cores=1,threads=1......  qemu     13224     1 13248   4 /usr/libexec/qemu-kvm -name test1 -S -M rhel6.4.0 -enable-kvm -m 1024 -smp 1,sockets=1,cores=1,threads=1......  qemu     13224     1 17191   0 /usr/libexec/qemu-kvm -name test1 -S -M rhel6.4.0 -enable-kvm -m 1024 -smp 1,sockets=1,cores=1,threads=1..... | |

大致的意思是，创建两个虚拟机centosb和test1的进程号分别是11535、13224。

将进程11535和13224分别绑定到CPU14（centosb）和CPU15（test1）上。在centosb上运行程序测试CPU性能。

进程测试之前的CPU情况：

|  |  |
| --- | --- |
| 1  2 | [root@test8 home]# mpstat -P ALL  Linux 2.6.32-279.2.1.el6.x86\_64 (test8.localdomain)     2013年12月17日  \_x86\_64\_        (16 CPU) |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | 21时01分14秒  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle  21时01分14秒  all    4.87    0.00    0.40    0.59    0.00    0.02    0.00    0.42   93.71  21时01分14秒    0   22.44    0.00    1.30    4.55    0.00    0.19    0.00    0.23   71.28  21时01分14秒    1   16.33    0.00    1.04    3.35    0.00    0.07    0.00    0.18   79.03  21时01分14秒    2    6.04    0.00    0.42    0.09    0.00    0.00    0.00    0.10   93.34  21时01分14秒    3    3.54    0.00    0.37    0.12    0.00    0.00    0.00    0.12   95.84  21时01分14秒    4    4.07    0.00    0.25    0.08    0.00    0.00    0.00    0.05   95.55  21时01分14秒    5    3.58    0.00    0.34    0.04    0.00    0.00    0.00    0.10   95.94  21时01分14秒    6    2.31    0.00    0.19    0.04    0.00    0.00    0.00    0.10   97.35  21时01分14秒    7    3.71    0.00    0.26    0.06    0.00    0.00    0.00    0.05   95.91  21时01分14秒    8    3.46    0.00    0.39    0.36    0.00    0.01    0.00    0.02   95.76  21时01分14秒    9    4.06    0.00    0.41    0.24    0.00    0.01    0.00    0.04   95.25  21时01分14秒   10    3.15    0.00    0.41    0.09    0.00    0.00    0.00    0.02   96.33  21时01分14秒   11    3.08    0.00    0.42    0.08    0.00    0.01    0.00    0.04   96.38  21时01分14秒   12    1.52    0.00    0.23    0.08    0.00    0.01    0.00    0.06   98.11  21时01分14秒   13    0.62    0.00    0.17    0.09    0.00    0.00    0.00    0.01   99.10  21时01分14秒   14    0.01    0.00    0.06    0.03    0.00    0.00    0.00    0.03   99.88  21时01分14秒   15    0.01    0.00    0.06    0.03    0.00    0.00    0.00    0.03   99.88 | |

CPU14和CPU15的情况一样，运行一段测试程序后的使用情况

|  |  |
| --- | --- |
| 1  2 | [root@test8 ~]# mpstat -P ALL  Linux 2.6.32-279.2.1.el6.x86\_64 (test8.localdomain)     2013年12月18日  \_x86\_64\_        (16 CPU) |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18 | 09时12分53秒  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle  09时12分53秒  all    4.74    0.00    0.35    0.44    0.00    0.02    0.00    0.18   94.26  09时12分53秒    0   25.30    0.00    1.33    4.11    0.00    0.23    0.00    0.03   69.01  09时12分53秒    1   15.11    0.01    0.90    2.33    0.00    0.06    0.00    0.02   81.57  09时12分53秒    2    5.39    0.01    0.37    0.03    0.00    0.00    0.00    0.01   94.19  09时12分53秒    3    3.42    0.00    0.31    0.03    0.00    0.00    0.00    0.01   96.22  09时12分53秒    4    3.86    0.00    0.23    0.02    0.00    0.00    0.00    0.01   95.89  09时12分53秒    5    2.96    0.00    0.20    0.01    0.00    0.00    0.00    0.01   96.82  09时12分53秒    6    2.19    0.00    0.17    0.01    0.00    0.00    0.00    0.01   97.61  09时12分53秒    7    3.33    0.00    0.21    0.01    0.00    0.00    0.00    0.01   96.44  09时12分53秒    8    3.03    0.00    0.34    0.20    0.00    0.00    0.00    0.00   96.42  09时12分53秒    9    3.75    0.01    0.38    0.11    0.00    0.00    0.00    0.00   95.75  09时12分53秒   10    2.53    0.00    0.30    0.02    0.00    0.00    0.00    0.00   97.15  09时12分53秒   11    2.64    0.00    0.30    0.02    0.00    0.00    0.00    0.00   97.03  09时12分53秒   12    1.48    0.00    0.18    0.01    0.00    0.01    0.00    0.01   98.30  09时12分53秒   13    0.91    0.01    0.18    0.02    0.00    0.00    0.00    0.00   98.89  09时12分53秒   14    0.01    0.00    0.11    0.10    0.00    0.00    0.00    2.64   97.14  09时12分53秒   15    0.02    0.00    0.14    0.09    0.00    0.00    0.00    0.10   99.65 | |

可以看到CPU14在%guest中的数据2.64是明显高于CPU15的%guest的0.10，如果你运行测试程序更长的时间，这个值会更高（为什么是%guest 这个值高，因为我是远程登录到测试主机，并运行程序的，所以是guest）

相关链接：

进程的处理器亲和性和vCPU的绑定：<http://smilejay.com/2012/08/kvm-vcpu-binding/>

查看物理CPU个数：<http://sundful.iteye.com/blog/1821856>

Linux下调节CPU使用的几种方法：<http://yunwei.org/123/>

如何找出CPU使用率：<http://www.51testing.com/html/20/n-218920-2.html>

[**Linux查看CPU和内存使用情况**](http://www.cnblogs.com/xd502djj/archive/2011/03/01/1968041.html)

在系统维护的过程中，随时可能有需要查看 CPU 使用率，并根据相应信息分析系统状况的需要。在 CentOS 中，可以通过 top 命令来查看 CPU 使用状况。运行 top 命令后，CPU 使用状态会以全屏的方式显示，并且会处在对话的模式 -- 用基于 top 的命令，可以控制显示方式等等。退出 top 的命令为 q （在 top 运行中敲 q 键一次）。

top命令是Linux下常用的性能分析工具，能够实时显示系统中各个进程的资源占用状况，类似于Windows的任务管理器

　　可以直接使用top命令后，查看%MEM的内容。可以选择按进程查看或者按用户查看，如想查看oracle用户的进程内存使用情况的话可以使用如下的命令：  
　　$ top -u oracle

内容解释：

　　PID：进程的ID  
　　USER：进程所有者  
　　PR：进程的优先级别，越小越优先被执行  
　　NInice：值  
　　VIRT：进程占用的虚拟内存  
　　RES：进程占用的物理内存  
　　SHR：进程使用的共享内存  
　　S：进程的状态。S表示休眠，R表示正在运行，Z表示僵死状态，N表示该进程优先值为负数  
　　%CPU：进程占用CPU的使用率  
　　%MEM：进程使用的物理内存和总内存的百分比  
　　TIME+：该进程启动后占用的总的CPU时间，即占用CPU使用时间的累加值。  
　　COMMAND：进程启动命令名称

　　操作实例:

　　在命令行中输入 “top”

　　即可启动 top

　　top 的全屏对话模式可分为3部分：系统信息栏、命令输入栏、进程列表栏。

　　第一部分 -- 最上部的 系统信息栏 ：

　　第一行（top）：

　　　　“00:11:04”为系统当前时刻；

　　　　“3:35”为系统启动后到现在的运作时间；

　　　　“2 users”为当前登录到系统的用户，更确切的说是登录到用户的终端数 -- 同一个用户同一时间对系统多个终端的连接将被视为多个用户连接到系统，这里的用户数也将表现为终端的数目；

　　　　“load average”为当前系统负载的平均值，后面的三个值分别为1分钟前、5分钟前、15分钟前进程的平均数，一般的可以认为这个数值超过 CPU 数目时，CPU 将比较吃力的负载当前系统所包含的进程；

　　第二行（Tasks）：

　　　　“59 total”为当前系统进程总数；

　　　　“1 running”为当前运行中的进程数；

　　　　“58 sleeping”为当前处于等待状态中的进程数；

　　　　“0 stoped”为被停止的系统进程数；

　　　　“0 zombie”为被复原的进程数；

　　第三行（Cpus）：

　　　　分别表示了 CPU 当前的使用率；

　　第四行（Mem）：

　　　　分别表示了内存总量、当前使用量、空闲内存量、以及缓冲使用中的内存量；

　　第五行（Swap）：

　　　　表示类别同第四行（Mem），但此处反映着交换分区（Swap）的使用情况。通常，交换分区（Swap）被频繁使用的情况，将被视作物理内存不足而造成的。

　　第二部分 -- 中间部分的内部命令提示栏：

　　top 运行中可以通过 top 的内部命令对进程的显示方式进行控制。内部命令如下表：

　　s

　　- 改变画面更新频率

　　l - 关闭或开启第一部分第一行 top 信息的表示

　　t - 关闭或开启第一部分第二行 Tasks 和第三行 Cpus 信息的表示

　　m - 关闭或开启第一部分第四行 Mem 和 第五行 Swap 信息的表示

　　N - 以 PID 的大小的顺序排列表示进程列表（第三部分后述）

　　P - 以 CPU 占用率大小的顺序排列进程列表 （第三部分后述）

　　M - 以内存占用率大小的顺序排列进程列表 （第三部分后述）

　　h - 显示帮助

　　n - 设置在进程列表所显示进程的数量

　　q - 退出 top

　　s -

　　改变画面更新周期

　　第三部分 -- 最下部分的进程列表栏：

　　以 PID 区分的进程列表将根据所设定的画面更新时间定期的更新。通过 top 内部命令可以控制此处的显示方式

**pmap**

可以根据进程查看进程相关信息占用的内存情况，(进程号可以通过ps查看)如下所示：  
　　$ pmap -d 5647

**ps**

　　如下例所示：  
　　$ ps -e -o 'pid,comm,args,pcpu,rsz,vsz,stime,user,uid'  其中rsz是是实际内存  
　　$ ps -e -o 'pid,comm,args,pcpu,rsz,vsz,stime,user,uid' | grep oracle |  sort -nrk

　　其中rsz为实际内存，上例实现按内存排序，由大到小

在Linux下查看内存我们一般用free命令：  
[root@scs-2 tmp]# free  
             total       used       free     shared    buffers     cached  
Mem:       3266180    3250004      16176          0     110652    2668236  
-/+ buffers/cache:     471116    2795064  
Swap:      2048276      80160    1968116

下面是对这些数值的解释：  
total:总计物理内存的大小。  
used:已使用多大。  
free:可用有多少。  
Shared:多个进程共享的内存总额。  
Buffers/cached:磁盘缓存的大小。  
第三行(-/+ buffers/cached):  
used:已使用多大。  
free:可用有多少。  
第四行就不多解释了。  
区别：第二行(mem)的used/free与第三行(-/+ buffers/cache) used/free的区别。 这两个的区别在于使用的角度来看，第一行是从OS的角度来看，因为对于OS，buffers/cached 都是属于被使用，所以他的可用内存是16176KB,已用内存是3250004KB,其中包括，内核（OS）使用+Application(X, oracle,etc)使用的+buffers+cached.  
第三行所指的是从应用程序角度来看，对于应用程序来说，buffers/cached 是等于可用的，因为buffer/cached是为了提高文件读取的性能，当应用程序需在用到内存的时候，buffer/cached会很快地被回收。  
所以从应用程序的角度来说，可用内存=系统free memory+buffers+cached。  
如上例：  
2795064=16176+110652+2668236

接下来解释什么时候内存会被交换，以及按什么方交换。 当可用内存少于额定值的时候，就会开会进行交换。  
如何看额定值：  
cat /proc/meminfo

[root@scs-2 tmp]# cat /proc/meminfo  
MemTotal:      3266180 kB  
MemFree:         17456 kB  
Buffers:        111328 kB  
Cached:        2664024 kB  
SwapCached:          0 kB  
Active:         467236 kB  
Inactive:      2644928 kB  
HighTotal:           0 kB  
HighFree:            0 kB  
LowTotal:      3266180 kB  
LowFree:         17456 kB  
SwapTotal:     2048276 kB  
SwapFree:      1968116 kB  
Dirty:               8 kB  
Writeback:           0 kB  
Mapped:         345360 kB  
Slab:           112344 kB  
Committed\_AS:   535292 kB  
PageTables:       2340 kB  
VmallocTotal: 536870911 kB  
VmallocUsed:    272696 kB  
VmallocChunk: 536598175 kB  
HugePages\_Total:     0  
HugePages\_Free:      0  
Hugepagesize:     2048 kB

用free -m查看的结果：  
[root@scs-2 tmp]# free -m   
             total       used       free     shared    buffers     cached  
Mem:          3189       3173         16          0        107       2605  
-/+ buffers/cache:        460       2729  
Swap:         2000         78       1921

查看/proc/kcore文件的大小（内存镜像）：  
[root@scs-2 tmp]# ll -h /proc/kcore   
-r-------- 1 root root 4.1G Jun 12 12:04 /proc/kcore

备注：

占用内存的测量

测量一个进程占用了多少内存，linux为我们提供了一个很方便的方法，/proc目录为我们提供了所有的信息，实际上top等工具也通过这里来获取相应的信息。

/proc/meminfo 机器的内存使用信息

/proc/pid/maps pid为进程号，显示当前进程所占用的虚拟地址。

/proc/pid/statm 进程所占用的内存

[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) 04

查看机器可用内存

/proc/28248/>free

total used free shared buffers cached

Mem: 1023788 926400 97388 0 134668 503688

-/+ buffers/cache: 288044 735744

Swap: 1959920 89608 1870312

我们通过free命令查看机器空闲内存时，会发现free的值很小。这主要是因为，在linux中有这么一种思想，内存不用白不用，因此它尽可能的cache和buffer一些数据，以方便下次使用。但实际上这些内存也是可以立刻拿来使用的。

所以 空闲内存=free+buffers+cached=total-used

top命令 是Linux下常用的性能 分析工具 ，能够实时显示系统 中各个进程的资源占用状况，类似于Windows的任务管理 器。下面详细介绍它的使用方法。

top - 02:53:32 up 16 days,  6:34, 17 users,  load average: 0.24, 0.21, 0.24  
Tasks: 481 total,   3 running, 474 sleeping,   0 stopped,   4 zombie  
Cpu(s): 10.3%us,  1.8%sy,  0.0%ni, 86.6%id,  0.5%wa,  0.2%hi,  0.6%si,  0.0%st  
Mem:   4042764k total,  4001096k used,    41668k free,   383536k buffers  
Swap:  2104472k total,     7900k used,  2096572k free,  1557040k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND  
32497 jacky     20   0  669m 222m  31m R   10  5.6       29:27.62 firefox  
 4788 yiuwing   20   0  257m  18m  13m S    5  0.5          5:42.44 konsole  
 5657 Liuxiaof  20   0  585m 159m  30m S    4  4.0          5:25.06 firefox  
 4455 xiefc      20   0  542m  124m  30m R    4  3.1         7:23.03 firefox  
 6188 Liuxiaof  20   0  191m   17m  13m S    4  0.5          0:01.16 konsole

统计信息区前五行是系统整体的统计信息。第一行是任务队列信息，同 uptime  命令的执行结果。其内容如下：

01:06:48  当前时间    
up 1:22  系统运行 时间，格式为时:分    
1 user  当前登录用户 数    
load average: 0.06, 0.60, 0.48  系统负载 ，即任务队列的平均长度。  
            三个数值分别为  1分钟、5分钟、15分钟前到现在的平均值。

第二、三行为进程和CPU的信息。当有多个CPU时，这些内容可能会超过两行。内容如下：

Tasks: 29 total  进程总数    
1 running  正在运行的进程数    
28 sleeping  睡眠的进程数    
0 stopped  停止的进程数    
0 zombie  僵尸进程数    
Cpu(s): 0.3% us  用户空间占用CPU百分比    
1.0% sy  内核 空间占用CPU百分比    
0.0% ni  用户进程空间内改变过优先级的进程占用CPU百分比    
98.7% id  空闲CPU百分比    
0.0% wa  等待输入输出的CPU时间百分比    
0.0% hi       
0.0% si

最后两行为内存 信息。内容如下：

Mem: 191272k total  物理内存总量    
173656k used  使用的物理内存总量    
17616k free  空闲内存总量    
22052k buffers  用作内核缓存 的内存量    
Swap: 192772k total  交换区总量    
0k used  使用的交换区总量    
192772k free  空闲交换区总量    
123988k cached  缓冲的交换区总量。  
            内存中的内容被换出到交换区，而后又被换入到内存，但使用过的交换区尚未被覆盖，  
            该数值即为这些内容已存在于内存中 的交换区的大小。  
            相应的内存再次被换出时可不必再对交换区写入。

进程信息区统计信息区域的下方显示了各个进程的详细信息。首先来认识一下各列的含义。

序号  列名  含义    
a  PID  进程id    
b  PPID  父进程id    
c  RUSER  Real user name    
d  UID  进程所有者的用户id    
e  USER  进程所有者的用户名    
f  GROUP  进程所有者的组名    
g  TTY  启动进程的终端名。不是从终端启动的进程则显示为 ?    
h  PR  优先级    
i  NI  nice值。负值表示高优先级，正值表示低优先级    
j  P  最后使用的CPU，仅在多CPU环境 下有意义    
k  %CPU  上次更新到现在的CPU时间占用百分比    
l  TIME  进程使用的CPU时间总计，单位秒    
m  TIME+  进程使用的CPU时间总计，单位1/100秒    
n  %MEM  进程使用的物理内存 百分比    
o  VIRT  进程使用的虚拟内存总量，单位kb。VIRT=SWAP+RES    
p  SWAP  进程使用的虚拟内存中，被换出的大小，单位kb。    
q  RES  进程使用的、未被换出的物理内存大小，单位kb。RES=CODE+DATA    
r  CODE  可执行代码占用的物理 内存大小，单位kb    
s  DATA  可执行代码以外的部分(数据 段+栈)占用的物理 内存大小，单位kb    
t  SHR  共享内存大小，单位kb    
u  nFLT  页面错误次数    
v  nDRT  最后一次写入到现在，被修改过的页面数。    
w  S  进程状态。  
            D =不可中断的睡眠状态  
            R =运行  
            S =睡眠  
            T =跟踪/停止  
            Z =僵尸进程    
x  COMMAND  命令名/命令行    
y  WCHAN  若该进程在睡眠，则显示睡眠中的系统函数名    
z  Flags  任务标志，参考 sched.h

默认情况下仅显示比较重要的  PID、USER、PR、NI、VIRT、RES、SHR、S、%CPU、%MEM、TIME+、COMMAND  列。可以通过下面的快捷键来更改显示内容。  
更改显示内容通过 f 键可以选择显示的内容。按 f 键之后会显示列的列表，按 a-z  即可显示或隐藏对应的列，最后按回车键确定。  
按 o 键可以改变列的显示顺序。按小写的 a-z 可以将相应的列向右移动，而大写的 A-Z  可以将相应的列向左移动。最后按回车键确定。  
按大写的 F 或 O 键，然后按 a-z 可以将进程按照相应的列进行排序。而大写的  R 键可以将当前的排序倒转。

==============================

top命令使用过程中，还可以使用一些交互的命令来完成其它参数的功能。这些命令是通过快捷键启动的。  
＜空格＞：立刻刷新。  
P：根据CPU使用大小进行排序。  
T：根据时间、累计时间排序。  
q：退出top命令。  
m：切换显示内存信息。  
t：切换显示进程和CPU状态信息。  
c：切换显示命令名称和完整命令行。  
M：根据使用内存大小进行排序。  
W：将当前设置写入~/.toprc文件中。这是写top配置文件的推荐方法。

可以看到，top命令是一个功能十分强大的监控系统的工具，对于系统管理员而言尤其重要。但是，它的缺点是会消耗很多系统资源。

 应用实例   
使用top命令可以监视指定用户，缺省情况是监视所有用户的进程。如果想查看指定用户的情况，在终端中按“U”键，然后输入用户名，系统就会切换为指定用户的进程运行界面。  
ａ.作用  
free命令用来显示内存的使用情况，使用权限是所有用户。  
ｂ.格式  
free [－b　－k　－m] [－o] [－s delay] [－t] [－V]  
ｃ.主要参数  
－b －k －m：分别以字节（KB、MB）为单位显示内存使用情况。  
－s delay：显示每隔多少秒数来显示一次内存使用情况。  
－t：显示内存总和列。  
－o：不显示缓冲区调节列。  
ｄ.应用实例  
free命令是用来查看内存使用情况的主要命令。和top命令相比，它的优点是使用简单，并且只占用很少的系统资源。通过－S参数可以使用free命令不间断地监视有多少内存在使用，这样可以把它当作一个方便实时监控器。  
＃free －b －s5  
使用这个命令后终端会连续不断地报告内存使用情况（以字节为单位），每5秒更新一次。