

进程，系统性能和计划任务 2

作者: [Carey](#)

原文链接: <https://ld246.com/article/1609831356596>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<p></p>

<p>2 进程管理和性能工具</p>

<p>参考链接: http://www.brendangregg.com/linuxperf.html</p>

<p></p>

<h2 id="2-1-进程树pstree">2.1 进程树 pstree</h2>

<p>pstree 可以用来显示进程的父子关系, 以树形结构显示</p>

<p>格式: </p>

<pre><code class="highlight-chroma">pstree [OPTION] [PID | USER]</code></pre>

<p>常用选项: </p>

<pre><code class="highlight-chroma">-p 显示PID</code></pre>

-T 不显示线程thead, 默认显示线程

-u 显示用户切换

-H pid 高亮显示
定进程及其前辈进程

</code></pre>

<p>范例: </p>

<pre><code class="highlight-chroma">[19:17:24 root@centos8 ~]#pstree 1</code>

[19:18:01 root@c
ntos8 ~]#pstree zhang

bash——ping

[19:18:47 root@c
ntos8 ~]#pstree -pT

[19:19:05 root@c
ntos8 ~]#pstree -u

</code></pre>

<p>2.2 进程信息 ps</p>

<p>ps 即 process state, 可以进程当前状态的快照, 默认显示当前终端中的进程, Linux 系统各进
的相关信息均保存在/proc/PID 目录下的各文件中</p>

<p>ps 格式</p>

<p>ps [OPTION]...</p>

<p>支持三种选项: </p>

UNIX 选项如: -A -e

BSD 选项如: a

GNU 选项如: --help

<p>常用选项: </p>

<pre><code class="highlight-chroma">a 选项包括所有终端中的进程x 选项包括不链接终端的进程u 选项显示进程所有者的信息

f 选项显示进程树,
当于 --forest

k|--sort 属性 对
性排序,属性前加 - 表示倒序

o 属性... 选项显示

制的信息 pid、cmd、%cpu、%mem L 显示支持的属性列表

-C cmdlist 指定命令, 多个命令用, 分隔

-L 显示线程

-e 显示所有进程相当于-A

-f 显示完整格式程信息

-F 显示更完整格式进程信息

-H 以进程层级格显示进程相关信息

-u userlist 指定效的用户ID或名称

-U userlist 指定真的用户ID或名称

-g gid或groupna e 指定有效的gid或组名称

-G gid或groupna e 指定真正的gid或组名称

-p pid 显示指pid进程

--ppid pid 显示于pid的子进程

-t ttylist 指定tty, 当于 t

-M 显示SELinux 息, 相当于Z

</code></pre>

<p>ps 输出属性</p>

<pre><code class="highlight-chroma">C: ps -ef 显示列 C 表示cpu利用率

VSZ: Virtual mem ry SiZe, 虚拟内存集, 线性内存RSS: ReSident Size, 常驻内存集

STAT: 进程状态

R: running

S: interruptable sleeping

D: uninterrupta le sleeping

T: stopped

Z: zombie

+: 前台进程

l: 多线程进程L: 内存分页并带锁

:

低优先级进程

<: 高优先级程

s: session leade, 会话(子进程)发起者

I: Idle kernel th ead, CentOS 8 新特性

ni: nice值

```

</span></span><span class="highlight-line"><span class="highlight-cl">pri: priority 优先级
</span></span><span class="highlight-line"><span class="highlight-cl">rtprio: 实时优先级
</span></span><span class="highlight-line"><span class="highlight-cl">psr: processor C
U编号
</span></span></code></pre>
<p><strong>示例: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">[19:19:25 root@centos8 ~]#ps axo pid,cmd,psr,ni,pri,rtprio
</span></span></code></pre>
<p><strong>常用组合</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">aux
</span></span><span class="highlight-line"><span class="highlight-cl">-ef
</span></span><span class="highlight-line"><span class="highlight-cl">-eFH
</span></span><span class="highlight-line"><span class="highlight-cl">-eo pid,tid,class,rt
rio,ni,pri,psr,pcpu,stat,comm
</span></span><span class="highlight-line"><span class="highlight-cl">axo stat,euid,ruid,t
y,tpgid,sess,pgrp,ppid,pid,pcpu,comm
</span></span></code></pre>
<p><strong>范例: 查看进程详细信息</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">[19:23:25 root@centos8 ~]#ps -ef
</span></span><span class="highlight-line"><span class="highlight-cl">[19:24:57 root@c
entos8 ~]#ps aux
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看进程特定属性
</span></span><span class="highlight-line"><span class="highlight-cl">[19:25:59 root@c
entos8 ~]#ps axo pid,cmd,%mem,%cpu
</span></span></code></pre>
<p><strong>范例: 针对属性排序, Centos6 以下版本不支持</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">#按CPU利用率倒序排序
</span></span><span class="highlight-line"><span class="highlight-cl">[19:27:36 root@c
entos8 ~]#ps aux k -%cpu
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#按内存倒序排序
</span></span><span class="highlight-line"><span class="highlight-cl">[19:29:46 root@c
entos8 ~]#ps axo pid,cmd,%cpu,%mem k -%mem
</span></span></code></pre>
<p><strong>范例: 有效用户和实际用户</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">[zhang@centos8 ~]$ passwd
</span></span><span class="highlight-line"><span class="highlight-cl">Changing passwo
d for user zhang.
</span></span><span class="highlight-line"><span class="highlight-cl">Current password:
</span></span><span class="highlight-line"><span class="highlight-cl">[19:30:34 root@c
entos8 ~]#ps axo pid,cmd,%cpu,%mem,user,euser,ruser | grep passwd
</span></span><span class="highlight-line"><span class="highlight-cl">1878 passwd
0.0 0.8 root root zhang
</span></span><span class="highlight-line"><span class="highlight-cl">1880 grep --color
auto passwd 0.0 0.1 root root root
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">euser: 为有效用
, 执行这个进程生效的用户, 一般执行文件带有suid权限会出现实际用户和有效用户不一样

```

```
</span></span><span class="highlight-line"><span class="highlight-cl">ruser: 为实际用
, 实际执行这个进程的用户
</span></span></code></pre>
<p><strong>范例: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">#查询你拥有的所有进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:38:23 root@c
ntos8 ~]#ps -x
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#显示指定用户名(
UID)或用户ID的进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:38:52 root@c
ntos8 ~]#ps -u postfix
</span></span><span class="highlight-line"><span class="highlight-cl">[19:39:22 root@c
ntos8 ~]#ps -u 89
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#显示指定用户名(E
ID)或用户ID的进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:39:54 root@c
ntos8 ~]#ps -fu postfix
</span></span><span class="highlight-line"><span class="highlight-cl">[19:39:29 root@c
ntos8 ~]#ps -fu 89
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看以root用户
限(实际和有效ID)运行的每个进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:40:36 root@c
ntos8 ~]#ps -U root -u root
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#列出某个组拥有
所有进程(实际组ID: RGID或名称)
</span></span><span class="highlight-line"><span class="highlight-cl">[19:40:36 root@c
ntos8 ~]#ps -fG postfix
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#列出有效组名称
或会话)所拥有的所有进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:41:53 root@c
ntos8 ~]#ps -fg postfix
</span></span><span class="highlight-line"><span class="highlight-cl">[19:42:37 root@c
ntos8 ~]#ps -fg 89
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#显示指定的进程I
对应的进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:43:30 root@c
ntos8 ~]#ps -fp 1853
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#以父进程ID来显
其下所有的进程, 如显示父进程为855的所有进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:45:04 root@c
ntos8 ~]#ps -f --ppid 855
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#显示指定PID的
个进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:46:42 root@c
ntos8 ~]#ps -fp 1,835
```



```
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#要按tty显示所属
程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:46:46 root@c
ntos8 ~]#ps -ft pts/0
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#以进程树显示系
中的进程如何相互链接
</span></span><span class="highlight-line"><span class="highlight-cl">[19:47:30 root@c
ntos8 ~]#ps -e --forest
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#以进程树显示指
的进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:48:18 root@c
ntos8 ~]#ps -e --forest -C sshd
</span></span><span class="highlight-line"><span class="highlight-cl">[19:48:18 root@c
ntos8 ~]#ps -e --forest | grep -v grep | grep sshd
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#要显示一个进程
所有线程,将显示LWP（轻量级进程）以及NLWP（轻量级进程数）列
</span></span><span class="highlight-line"><span class="highlight-cl">[19:50:05 root@c
ntos8 ~]#ps -fL -C nginx
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#要列出所有格式
明符
</span></span><span class="highlight-line"><span class="highlight-cl">[19:50:23 root@c
ntos8 ~]#ps -L
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看进程的PID,
PID, 用户名和命令
</span></span><span class="highlight-line"><span class="highlight-cl">[19:51:14 root@c
ntos8 ~]#ps -eo pid,ppid,user,cmd
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#自定义格式显示
件系统组,ni值开始时间和进程的时间
</span></span><span class="highlight-line"><span class="highlight-cl">[19:51:28 root@c
ntos8 ~]#ps -p 1 -o pid,ppid,fgroup,ni,lstart,etime
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#使用其PID查找
程名称:
</span></span><span class="highlight-line"><span class="highlight-cl">[19:52:06 root@c
ntos8 ~]#ps -p 1830 -o comm=
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#要以其名称选择
定进程, 显示其所有子进程
</span></span><span class="highlight-line"><span class="highlight-cl">[19:52:50 root@c
ntos8 ~]#ps -C sshd,bash
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查找指定进程名
有的所属PID, 在编写需要从std输出或文件读取PID的脚本时这个参数很有用
</span></span><span class="highlight-line"><span class="highlight-cl">[19:53:11 root@c
ntos8 ~]#ps -C sshd -o pid=
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#检查一个进程的
```

行时间

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:53:38 root@centos8 ~]#ps -eo comm,etime,user | grep nginx
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">#排序，查找占用多内存和CPU的进程
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:54:06 root@centos8 ~]#ps -eo pid,ppid,cmd,%cpu,%mem --sort=-%mem | head
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:55:04 root@centos8 ~]#ps -eo pid,ppid,cmd,%cpu,%mem --sort=-%cpu | head
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">#显示安全信息
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:55:38 root@centos8 ~]#ps -eM
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:55:50 root@centos8 ~]#ps --context
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">#使用以下命令以户定义的格式显示安全信息
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:56:51 root@centos8 ~]#ps -eo euser,ruser,suser,fuser,f,comm,label
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">#使用watch实用序执行重复的输出以实现对就程进行实时的监视，如下面的命令显示每秒钟的监视
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[19:58:12 root@centos8 ~]#watch -n 1 'ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head
```

```
</span></span></code></pre>
```

<h6 id="面试题-查看未知进程的执行程序文件路径">面试题：查看未知进程的执行程序文件路径</h6>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:00:27 root@centos8 ~]#ls -l /proc/1852/exe
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">lrwxrwxrwx 1 root oot 0 Jan  4 19:31 /proc/1852/exe -&gt; /usr/bin/su
```

```
</span></span></code></pre>
```

<p>范例：查看优先级和 CPU 绑定关系</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:02:06 root@centos8 ~]#ps axo pid,cmd,ni,pri,psr,rtprio | grep bash
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">1363 -bash
    0 19 0  -
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">1830 -bash
    0 19 0  -
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">1853 -bash
    0 19 0  -
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">4324 grep --color auto bash    0 19 0  -
```

```
</span></span></code></pre>
```

<p>范例：实现进程和 CPU 的绑定</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:04:14 root@centos8 ~]#taskset --help
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">Usage: taskset [options] [mask | cpu-list] [pid|cmd [args...]]
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">#并不能固定永久定，只要pid号变化就无效了
```

```
</span></span></code></pre>
```

2.3 查看进程信息 prtstat

可以显示进程信息,来自于 psmisc 包

格式:

```
prtstat [options] PID ...
```

选项:

-r raw 格式显示

```
[20:06:56 root@centos8 ~]#prtstat -r 4325
[20:07:38 root@centos8 ~]#prtstat 4325
```

2.4 设置和调整进程优先级

进程优先级调整

静态优先级: 100-139

进程默认启动时的 nice 值为 0, 优先级为 120

只有根用户才能降低 nice 值 (提高优先性)

nice 命令

以指定的优先级来启动进程

```
nice [OPTION] [COMMAND [ARG]...]
-n, --adjustment N  add integer N to the niceness (default 10)
```

renice 命令

可以调整正在执行中的进程的优先级

```
renice [-n] priority pid...
```

查看

```
ps axo pid,comm,ni
```

范例:

```
[20:08:03 root@centos8 ~]#nice -n -10 ping 127.0.0.1
[20:10:24 root@centos8 ~]#ps axo pid,cmd,nice | grep ping
4409 ping 127.0.0.1 -10
4415 grep --color auto ping 0
[20:11:35 root@centos8 ~]#renice -n 20 4417
4417 (process ID) ld priority -10, new priority 19
[20:11:46 root@centos8 ~]#ps axo pid,cmd,nice | grep ping
4417 ping 127.0.0.1 19
```



```
</span></span><span class="highlight-line"><span class="highlight-cl">4422 grep --color
auto ping      0
</span></span></code></pre>
<h2 id="2-5-搜索进程">2.5 搜索进程</h2>
<p><strong>按条件搜索进程</strong></p>
<ul>
<li>ps 选项 | grep 'pattern' 灵活</li>
<li>pgrep 按预定义的模式</li>
<li>/sbin/pidof 按确切的程序名称查看 pid</li>
</ul>
<h3 id="2-5-1-pgrep">2.5.1 pgrep</h3>
<p><strong>命令格式</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">pgrep [options] pattern
</span></span></code></pre>
<p><strong>常用选项</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">-u uid: effective user, 生效者
</span></span><span class="highlight-line"><span class="highlight-cl">-U uid: real user
真正发起运行命令者
</span></span><span class="highlight-line"><span class="highlight-cl">-t terminal: 与指
终端相关的进程
</span></span><span class="highlight-line"><span class="highlight-cl">-l: 显示进程名
</span></span><span class="highlight-line"><span class="highlight-cl">-a: 显示完整格式
进程名
</span></span><span class="highlight-line"><span class="highlight-cl">-P pid: 显示指定
程的子进程
</span></span></code></pre>
<p><strong>范例: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">[20:14:23 root@centos8 ~]#pgrep -u zhang
</span></span><span class="highlight-line"><span class="highlight-cl">4427
</span></span><span class="highlight-line"><span class="highlight-cl">[20:14:29 root@c
entos8 ~]#pgrep -lu zhang
</span></span><span class="highlight-line"><span class="highlight-cl">4427 bash
</span></span><span class="highlight-line"><span class="highlight-cl">[20:14:36 root@c
entos8 ~]#pgrep -au zhang
</span></span><span class="highlight-line"><span class="highlight-cl">[20:14:49 root@c
entos8 ~]#pgrep -aP 4427
</span></span><span class="highlight-line"><span class="highlight-cl">4455 dd if=/dev/z
ro of=/dev/null
</span></span><span class="highlight-line"><span class="highlight-cl">[20:15:27 root@c
entos8 ~]#pgrep -at pts/1
</span></span><span class="highlight-line"><span class="highlight-cl">1830 -bash
</span></span><span class="highlight-line"><span class="highlight-cl">4426 su - zhang
</span></span><span class="highlight-line"><span class="highlight-cl">4427 -bash
</span></span><span class="highlight-line"><span class="highlight-cl">4455 dd if=/dev/z
ro of=/dev/null
</span></span><span class="highlight-line"><span class="highlight-cl">[20:15:47 root@c
entos8 ~]#
</span></span></code></pre>
<h3 id="2-5-2-pidof">2.5.2 pidof</h3>
<p><strong>命令格式: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
```

```
cl">pidof [options] [program [...]]
</span></span></code></pre>
<p><strong>常用选项</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">-x 按脚本名称查找pid
</span></span></code></pre>
<p><strong>范例:</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:15:47 root@centos8 ~]#pidof bash
</span></span><span class="highlight-line"><span class="highlight-cl">4476 1830 1363
</span></span><span class="highlight-line"><span class="highlight-cl">[20:17:39 root@c
ntos8 ~]#pidof 1.sh
</span></span><span class="highlight-line"><span class="highlight-cl">[20:17:43 root@c
ntos8 ~]#pidof -x 1.sh
</span></span><span class="highlight-line"><span class="highlight-cl">4476
</span></span></code></pre>
<h2 id="2-6-负载查询-uptime">2.6 负载查询 uptime</h2>
<p><strong>/proc/uptime 包括两个值, 单位 s</strong></p>
<ul>
<li>系统启动时长</li>
<li>空闲进程的总时长 (按总的 CPU 核数计算) </li>
</ul>
<p><strong>uptime 和 w 显示以下内容</strong></p>
<ul>
<li>当前时间</li>
<li>系统已启动的时间</li>
<li>当前上线人数</li>
<li>系统平均负载 (1、5、15 分钟的平均负载, 一般不会超过 1, 超过 5 时建议警报) </li>
</ul>
<p>系统平均负载: 指在特定时间间隔内运行队列中的平均进程数,通常每个 CPU 内核的当前活动进
数不大于 3, 那么系统的性能良好。如果每个 CPU 内核的任务数大于 5, 那么此主机的性能有严重
题</p>
<p>如: linux 主机是 1 个双核 CPU, 当 Load Average 为 6 的时候说明机器已经被充分使用</p>
<p><strong>范例:</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:18:04 root@centos8 ~]#uptime
</span></span><span class="highlight-line"><span class="highlight-cl">20:19:22 up 2:33,
2 users, load average: 0.10, 0.42, 0.42
</span></span><span class="highlight-line"><span class="highlight-cl">[20:19:22 root@c
ntos8 ~]#w
</span></span><span class="highlight-line"><span class="highlight-cl">20:19:27 up 2:33,
2 users, load average: 0.09, 0.41, 0.42
</span></span><span class="highlight-line"><span class="highlight-cl">USER    TTY      F
OM          LOGIN@  IDLE   JCPU   PCPU WHAT
</span></span><span class="highlight-line"><span class="highlight-cl">root    pts/0    192
168.10.1   19:07   1:40   0.29s  0.29s -bash
</span></span><span class="highlight-line"><span class="highlight-cl">root    pts/1    192
168.10.1   19:31   6.00s  0.03s  0.00s w
</span></span></code></pre>
<h2 id="2-7-显示CPU相关统计mpstat">2.7 显示 CPU 相关统计 mpstat</h2>
<p>来自于 sysstat 包</p>
<p><strong>范例:</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:20:44 root@centos8 ~]#mpstat
```

```

</span></span><span class="highlight-line"><span class="highlight-cl">Linux 4.18.0-193.e
8.x86_64 (centos8) 01/04/2021 x86_64 (1 CPU)
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">08:20:48 PM CPU
%usr %nice %sys %iowait %irq %soft %steal %guest %gnice %idle
</span></span><span class="highlight-line"><span class="highlight-cl">08:20:48 PM all
3.36 0.02 3.17 0.08 0.14 0.08 0.00 0.00 0.00 93.15
</span></span><span class="highlight-line"><span class="highlight-cl">[20:20:48 root@c
entos8 ~]#mpstat 1 3
</span></span><span class="highlight-line"><span class="highlight-cl">Linux 4.18.0-193.e
8.x86_64 (centos8) 01/04/2021 x86_64 (1 CPU)
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">08:20:59 PM CPU
%usr %nice %sys %iowait %irq %soft %steal %guest %gnice %idle
</span></span><span class="highlight-line"><span class="highlight-cl">08:21:00 PM all
0.00 0.00 0.00 0.00 1.00 0.00 0.00 0.00 0.00 99.00
</span></span><span class="highlight-line"><span class="highlight-cl">08:21:01 PM all
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00
</span></span><span class="highlight-line"><span class="highlight-cl">08:21:02 PM all
0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 100.00
</span></span><span class="highlight-line"><span class="highlight-cl">Average: all 0
00 0.00 0.00 0.00 0.33 0.00 0.00 0.00 0.00 99.67
</span></span></code></pre>
<h2 id="2-8-查看进程实时状态top和htop">2.8 查看进程实时状态 top 和 htop</h2>
<h3 id="2-8-1-top">2.8.1 top</h3>
<p></p>
<p><strong>top 提供动态的实时进程状态</strong></p>
<p>有许多内置命令</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">帮助: h 或 ? , 按 q 或esc 退出帮助
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">排序:
</span></span><span class="highlight-line"><span class="highlight-cl">P: 以占据的CPU
分比,%CPU
</span></span><span class="highlight-line"><span class="highlight-cl">M: 占据内存百分比
%MEM
</span></span><span class="highlight-line"><span class="highlight-cl">T: 累积占据CPU
长,TIME+
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">首部信息显示:
</span></span><span class="highlight-line"><span class="highlight-cl">uptime信息: l命令
</span></span><span class="highlight-line"><span class="highlight-cl">tasks及cpu信息:
命令
</span></span><span class="highlight-line"><span class="highlight-cl">cpu分别显示: 1 (
字)
</span></span><span class="highlight-line"><span class="highlight-cl">memory信息: m
令
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">退出命令: q
</span></span><span class="highlight-line"><span class="highlight-cl">修改刷新时间间隔
s
</span></span><span class="highlight-line"><span class="highlight-cl">终止指定进程: k
</pre>

```

```

</span></span><span class="highlight-line"><span class="highlight-cl">保存文件: W
</span></span></code></pre>
<p><strong>top 命令栏位信息简介</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight"
cl">us: 用户空间
</span></span><span class="highlight-line"><span class="highlight-cl">sy: 内核空间
</span></span><span class="highlight-line"><span class="highlight-cl">ni: 调整nice时间
</span></span><span class="highlight-line"><span class="highlight-cl">id: 空闲
</span></span><span class="highlight-line"><span class="highlight-cl">wa: 等待IO时间
</span></span><span class="highlight-line"><span class="highlight-cl">hi: 硬中断
</span></span><span class="highlight-line"><span class="highlight-cl">si: 软中断 (模式
换)
</span></span><span class="highlight-line"><span class="highlight-cl">st: 虚拟机偷走的
间
</span></span></code></pre>
<p><strong>top 选项</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight"
cl">-d # 指定刷新实际间隔, 默认为3秒
</span></span><span class="highlight-line"><span class="highlight-cl">-b 全部显示所
进程
</span></span><span class="highlight-line"><span class="highlight-cl">-n # 刷新多少次
退出
</span></span><span class="highlight-line"><span class="highlight-cl">-H 线程模式
</span></span></code></pre>
<p><strong>示例: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight"
cl">[20:28:03 root@centos8 ~]#top -H -p 855
</span></span></code></pre>
<h3 id="2-8-2-htop">2.8.2 htop</h3>
<p>htop 命令是增强版的 TOP 命令, 来自 EPEL 源, 比 top 功能更强<br>
 <
p>
<p><strong>选项: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight"
cl">-d #: 指定延迟时间;
</span></span><span class="highlight-line"><span class="highlight-cl">-u UserName: 仅
示指定用户的进程
</span></span><span class="highlight-line"><span class="highlight-cl">-s COLUME: 以指
字段进行排序
</span></span></code></pre>
<p><strong>子命令: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight"
cl">s: 跟踪选定进程的系统调用
</span></span><span class="highlight-line"><span class="highlight-cl">l: 显示选定进程打
的文件列表
</span></span><span class="highlight-line"><span class="highlight-cl">a: 将选定的进程
定至某指定CPU核心
</span></span><span class="highlight-line"><span class="highlight-cl">t: 显示进程树
</span></span></code></pre>
<h2 id="2-9-内存空间free">2.9 内存空间 free</h2>
<p> </p>

```

向 /proc/sys/vm/drop_caches 中写入相应的修改值，会清理缓存。建议先执行 sync (sync 命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)。行 echo 1、2、3 至 /proc/sys/vm/drop_caches, 达到不同的清理目的

如果因为应用有像内存泄露、溢出的问题时，从 swap 的使用情况是可以比较快速可以判断的但通过执行 free 反而比较难查看。但核心并不会因为内存泄露等问题并没有快速清空 buffer 或 cach (默认值是 0)，生产也不应该随便去改变此值。

一般情况下，应用在系统上稳定运行了，free 值也会保持在一个稳定值的。当发生内存不足、应获取不到可用内存、OOM 错误等问题时，还是更应该去分析应用方面的原因，否则，清空 buffer 强制腾出 free 的大小，可能只是把问题给暂时屏蔽了。

排除内存不足的情况外，除非是在软件开发阶段，需要临时清掉 buffer，以判断应用的内存使用情况；或应用已经不再提供支持，即使应用对内存的时候确实有问题，而且无法避免的情况下，才考虑时清空 buffer。

说明: man 5 proc

```
<code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[root@centos8 ~]#man proc
</span></span><span class="highlight-line"><span class="highlight-cl">.....
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">To free pagecache
use:
</span></span><span class="highlight-line"><span class="highlight-cl">echo 1 &gt; /proc
sys/vm/drop_caches
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">To free dentries a
d inodes, use:
</span></span><span class="highlight-line"><span class="highlight-cl">echo 2 &gt; /proc
sys/vm/drop_caches
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">To free pagecache
dentries and inodes, use:
</span></span><span class="highlight-line"><span class="highlight-cl">echo 3 &gt; /proc
sys/vm/drop_caches
</span></span></code></pre>
```

范例：清理缓存

```
<code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:32:40 root@centos8 ~]#cat /proc/sys/vm/drop_caches
</span></span><span class="highlight-line"><span class="highlight-cl">0
</span></span><span class="highlight-line"><span class="highlight-cl">[20:32:52 root@c
entos8 ~]#free -h
</span></span><span class="highlight-line"><span class="highlight-cl">total      used
free   shared buff/cache available
</span></span><span class="highlight-line"><span class="highlight-cl">Mem:      952Mi
202Mi    353Mi    6.0Mi    395Mi    590Mi
</span></span><span class="highlight-line"><span class="highlight-cl">Swap:     2.0Gi
0B       2.0Gi
</span></span><span class="highlight-line"><span class="highlight-cl">[20:32:58 root@c
entos8 ~]#echo 3 &gt; /proc/sys/vm/drop_caches
</span></span><span class="highlight-line"><span class="highlight-cl">[20:33:19 root@c
entos8 ~]#free -h
</span></span><span class="highlight-line"><span class="highlight-cl">total      used
free   shared buff/cache available
</span></span><span class="highlight-line"><span class="highlight-cl">Mem:      952Mi
181Mi    649Mi    6.0Mi    121Mi    636Mi
</span></span><span class="highlight-line"><span class="highlight-cl">Swap:     2.0Gi
0B       2.0Gi
</span></span></code>
```


</code></pre>

<h2 id="2-10-进程对应的内存映射pmap">2.10 进程对应的内存映射 pmap</h2>

<p>格式:</p>

<pre><code class="highlight-chroma">pmap [options] pid [...]

</code></pre>

<p>常用选项</p>

<pre><code class="highlight-chroma">-x: 显示详细格式的信息

</code></pre>

<p>范例:</p>

<pre><code class="highlight-chroma">[20:33:20 root@centos8 ~]#pmap 1

[20:35:04 root@c

ntos8 ~]#cat /proc/1/maps

</code></pre>

<p>范例: 查看系统调用与库调用</p>

<pre><code class="highlight-chroma">#系统调用

[20:36:16 root@c

ntos8 ~]#strace ls

#库调用

[20:37:01 root@c

ntos8 ~]#ltrace ls

</code></pre>

<h2 id="2-11-虚拟内存信息vmstat">2.11 虚拟内存信息 vmstat</h2>

<p>格式:</p>

<pre><code class="highlight-chroma">vmstat [options] [delay [count]]

</code></pre>

<p>显示项说明</p>

<pre><code class="highlight-chroma">procs:

r: 可运行 (正

行或等待运行) 进程的个数, 和核心数有关

b: 处于不可中

睡眠态的进程个数(被阻塞的队列的长度)

memory:

>swpd: 交换内存

使用总量

>free: 空闲物理

内存总量

>buffer: 用于buf

er的内存总量

>cache: 用于cac

e的内存总量

>swap:

>si: 从磁盘交换

内存的数据速率(kb/s)

>so: 从内存交换

磁盘的数据速率(kb/s)

>io:

>bi: 从块设备读

数据到系统的速率(kb/s)

` bo: 保存数据至设备的速率`

`system:`

` in: interrupts
中断速率, 包括时钟`

` cs: context switch
进程切换速率`

`cpu:`

` us:Time spent running non-kernel code`

` sy: Time spent running kernel code`

` id: Time spent idle. Linux 2.5.41前,包括IO-wait time.`

` wa: Time spent waiting for IO. 2.5.41前, 包括in idle.`

` st: Time stolen from a virtual machine. 2.6.11前, unknown.`

`</code></pre>`

`<p>选项: </p>`

`<pre><code class="highlight-chroma">-s 显示内存的统计数据`

`</code></pre>`

`<p>范例: </p>`

`<pre><code class="highlight-chroma">>[20:39:42 root@centos8 ~]#vmstat`

`procs -----memory----- --swap-- -----io---- -system-- -----cpu-----`

`r b swpd free buff cache si so bi bo in cs us sy id wa st`

`2 0 0 495288 72 276680 0 0 41 27 113 129 3 3 94 0 0`

`>[20:39:46 root@centos8 ~]#vmstat 1 3`

`procs -----memory----- --swap-- -----io---- -system-- -----cpu-----`

`r b swpd free buff cache si so bi bo in cs us sy id wa st`

`2 0 0 495288 72 276700 0 0 41 27 113 129 3 3 94 0 0`

`0 0 0 495228 72 276700 0 0 0 0 60 109 0 0 100 0 0`

`0 0 0 495228 72 276700 0 0 0 0 58 110 0 0 100 0 0`

`>[20:39:54 root@centos8 ~]#vmstat -s`

`974876 K total memory`

`202756 K used memory`

`210360 K active memory`

`83320 K inactive`

```

emory
</span></span><span class="highlight-line"><span class="highlight-cl">495348 K free m
mory
</span></span><span class="highlight-line"><span class="highlight-cl">72 K buffer memo
y
</span></span><span class="highlight-line"><span class="highlight-cl">276700 K swap ca
he
</span></span><span class="highlight-line"><span class="highlight-cl">2097148 K total s
ap
</span></span><span class="highlight-line"><span class="highlight-cl">0 K used swap
</span></span><span class="highlight-line"><span class="highlight-cl">2097148 K free s
ap
</span></span><span class="highlight-line"><span class="highlight-cl">31561 non-nice u
er cpu ticks
</span></span><span class="highlight-line"><span class="highlight-cl">213 nice user cpu
icks
</span></span><span class="highlight-line"><span class="highlight-cl">29939 system cpu
icks
</span></span><span class="highlight-line"><span class="highlight-cl">978883 idle cpu ti
ks
</span></span><span class="highlight-line"><span class="highlight-cl">889 IO-wait cpu ti
ks
</span></span><span class="highlight-line"><span class="highlight-cl">1465 IRQ cpu ticks
</span></span><span class="highlight-line"><span class="highlight-cl">877 softirq cpu tic
s
</span></span><span class="highlight-line"><span class="highlight-cl">0 stolen cpu ticks
</span></span><span class="highlight-line"><span class="highlight-cl">427309 pages pa
ed in
</span></span><span class="highlight-line"><span class="highlight-cl">284609 pages pa
ed out
</span></span><span class="highlight-line"><span class="highlight-cl">0 pages swapped
n
</span></span><span class="highlight-line"><span class="highlight-cl">0 pages swapped
ut
</span></span><span class="highlight-line"><span class="highlight-cl">1182780 interrupt

</span></span><span class="highlight-line"><span class="highlight-cl">1347479 CPU con
ext switches
</span></span><span class="highlight-line"><span class="highlight-cl">1609753548 boot
ime
</span></span><span class="highlight-line"><span class="highlight-cl">6035 forks
</span></span></code></pre>
<h2 id="2-12-统计CPU和设备IO信息iostat">2.12 统计 CPU 和设备 IO 信息 iostat</h2>
<p><strong>iostat 可以提供更丰富的 IO 性能状态数据</strong></p>
<p>此工具由 sysstat 包提供</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">常用选项:
</span></span><span class="highlight-line"><span class="highlight-cl">-c 只显示CPU行
</span></span><span class="highlight-line"><span class="highlight-cl">-d 显示设备〈磁盘
使用状态
</span></span><span class="highlight-line"><span class="highlight-cl">-k 以千字节为为单
显示输出
</span></span><span class="highlight-line"><span class="highlight-cl">-t 在输出中包括时
戳



原文链接: 进程, 系统性能和计划任务 2


```

```

</span></span><span class="highlight-line"><span class="highlight-cl">-x 在输出中包括扩
的磁盘指标
</span></span></code></pre>
<p><strong>范例: </strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:42:19 root@centos8 ~]#iostat 1 3
</span></span><span class="highlight-line"><span class="highlight-cl">Linux 4.18.0-193.e
8.x86_64 (centos8) 01/04/2021 x86_64 (1 CPU)
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">avg-cpu: %user
%nice %system %iowait %steal %idle
</span></span><span class="highlight-line"><span class="highlight-cl">2.98 0.02 3.06
0.08 0.00 93.85
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">Device tps
kB_read/s kB_wrtn/s kB_read kB_wrtn
</span></span><span class="highlight-line"><span class="highlight-cl">sda 1.63
40.34 26.87 427409 284764
</span></span><span class="highlight-line"><span class="highlight-cl">dm-0 1.68
39.25 26.87 415920 284752
</span></span><span class="highlight-line"><span class="highlight-cl">dm-1 0.01
0.21 0.00 2220 0
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">avg-cpu: %user
%nice %system %iowait %steal %idle
</span></span><span class="highlight-line"><span class="highlight-cl">0.00 0.00 0.00
0.00 0.00 100.00
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">Device tps
kB_read/s kB_wrtn/s kB_read kB_wrtn
</span></span><span class="highlight-line"><span class="highlight-cl">sda 0.00
0.00 0.00 0 0
</span></span><span class="highlight-line"><span class="highlight-cl">dm-0 0.00
0.00 0.00 0 0
</span></span><span class="highlight-line"><span class="highlight-cl">dm-1 0.00
0.00 0.00 0 0
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">avg-cpu: %user
%nice %system %iowait %steal %idle
</span></span><span class="highlight-line"><span class="highlight-cl">0.00 0.00 0.00
0.00 0.00 100.00
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">Device tps
kB_read/s kB_wrtn/s kB_read kB_wrtn
</span></span><span class="highlight-line"><span class="highlight-cl">sda 0.00
0.00 0.00 0 0
</span></span><span class="highlight-line"><span class="highlight-cl">dm-0 0.00
0.00 0.00 0 0
</span></span><span class="highlight-line"><span class="highlight-cl">dm-1 0.00
0.00 0.00 0 0
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">tps: 该设备每秒
传输次数 (Indicate the number of transfers per second that were
</span></span><span class="highlight-line"><span class="highlight-cl">issued to the devi

```

e.)。"一次传输"意思是"一次I/O请求"。多个逻辑请求可能会被合并为"一次I/O

请求"。"一次传输请求的大小是未知的。

kB_read/s: 每秒设备 (drive expressed) 读取的数据量;

kB_wrtn/s: 每秒设备 (drive expressed) 写入的数据量;

kB_read: 读取的总数据量;

kB_wrtn: 写入的数量数据量; 这些单位都为Kilobytes。

范例:

```
[20:44:09 root@centos8 ~]#iostat -d sda -x
```

```
Linux 4.18.0-193.el8.x86_64 (centos8) 01/04/2021 _x86_64_ (1 CPU)
```

```
Device r/s w/s kB/s kB/s rrqm/s wrqm/s %rrqm %wrqm r_await w_await aqu-sz rareq-sz warq-sz svctm %util
```

```
sda 1.06 0.55 39.92 26.60 0.01 0.08 0.64 12.94 0.94 0.51 0.00 37.62 48.31 0.43 0.7
```

r/s: 每秒合并后读请求数w/s: 每秒合并后写的请求数rsec/s: 每秒读取的扇区数; wsec/: 每秒写入的扇区数。

rKB/s: The number of read requests that were issued to the device per second; wKB/s: The number of write requests that were issued to the device per second; rrqm/s: 每秒这个设备相关的读取请求有多少被Merge了 (当系统调用需要读取数据的时候, VFS将请求发到 各个FS, 如果FS发现不同的读取请求读取的是相同Block的数据, FS会将这个请求合并Merge) ; wrqm/s: 每秒这个设备相关的写入请求有多少被Merge了。

%rrqm: The percentage of read requests merged together before being sent to the device.

%wrqm: The percentage of write requests merged together before being sent to the device.

avgrq-sz 平均请求扇区的大小

avgqu-sz 是平均请求队列的长度。毫无疑问, 队列长度越短越好。

await: 每一个I/O请求的处理的平均时间 (单位是微秒毫秒)。这里可以理解为IO的响应时间, 一般地系统IO响应时应该低于5ms, 如果大于10ms就比较大了。这个时间包括了队列时间和服务时间, 也就是说, 一般情况下, await大于svctm, 它们的差值越小, 则说明队列时间越短, 反之差值越大, 队列时间越长, 说系统出了问题。

svctm 表示平均每次设备I/O操作的服务时间 (以毫秒为单位)。如果svctm的值与await很接近, 表示 几乎没有I/O等, 磁盘性能很好, 如果await的值远高于svctm的值, 则表示I/O队列等待太长, 系统上运行的应用程序将变慢。

%util: 在统计时间内所有处理IO时间, 除以总共统计时间。例如, 如果统计间隔1秒, 该设备有0.8秒在处理IO, 而0.2闲置, 那么该设备的%util = 0.8/1 = 80%, 所以该参数暗示了设备的繁忙程度。一般地, 如果该参

是100%表示设备已经接近满负荷运行了（当然如果是多磁盘，即使%util是100%，因为磁盘 的并发力，所以磁盘使用未必就到了瓶颈）。

```
</span></span></code></pre>
```

<h2 id="2-13-监视磁盘I-O-iotop">2.13 监视磁盘 I/O iotop</h2>

<p>来自于 iotop 包</p>

<p></p>

<p>iotop 命令是一个用来监视磁盘 I/O 使用状况的 top 类工具 iotop 具有与 top 相似的 UI，其中包括 PID、用 户、I/O、进程等相关信息，可查看每个进程是如何使用 IO</p>

<p>iotop 输出</p>

第一行：Read 和 Write 速率总计

第二行：实际的 Read 和 Write 速率

第三行：参数如下：

线程 ID（按 p 切换为进程 ID）

优先级

用户

磁盘读速率

磁盘写速率

swap 交换百分比

IO 等待所占的百分比

<p>iotop 常用参数</p>

<pre><code class="highlight-chroma">-o, --only只显示正在产生I/O的进程或线程，除了传参，可以在运行过程中按o生效-b, --batch非交互式，一般用来记录日志-n NUM, --iter=N M设置监测的次数，默认无限。在非交互模式下很有用-d SEC, --delay=S C设置每次监测的间隔，默认1秒，接受非整形数据例如1.1-p PID, --pid=PI 指定监测的进程/线程-u USER, --user=SER指定监测某个用户产生的I/O-P, --processes仅示进程，默认iotop显示所有线程-a, --accumulate 显示累积的I/O，而不是带宽-k, --kilobytes使用B单位，而不是对人友好的单位。在非交互模式下，脚本编程有用-t, --time 加上时戳，非交互非模式-q, --quiet 禁止头行，非交互模式，有三种指定方式-q 只在第一次监时显示列名-qq 永远不显示列-qqq 永远不显示I/

汇总

```
</span></span></code></pre>
```

<p>交互按键</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">left和right方向键：改变排序
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">r：反向排序
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">o：切换至选项--only
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">p：切换至--processes选项
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">a：切换至--accumulated选项
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">q：退出
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">i：改变线程的优先
```


```
</span></span></code></pre>
```

<p>通过 EPEL 源的 iftop 包安装</p> ``` <pre><code class="highlight-chroma">[20:50:46 root@centos8 ~]#iftop -ni eth0 ``` ``` </code></pre> ``` <p></p> <p>nload 是一个实时监控网络流量和带宽使用情况，以数值和动态图展示进出的流量情况,通过 EPEL 源安装</p> <p>界面操作</p> ``` <pre><code class="highlight-chroma">上下方向键、左右方向键、enter键或者tab键都可以切换查看多个网卡的流量情况 ``` ``` 按 F2 显示选项窗口 ``` ``` 按 q 或者 Ctrl+C 出 nload ``` ``` </code></pre> ``` <p>范例：</p> ``` <pre><code class="highlight-chroma">#默认只查看第一个网络的流量进出情况 ``` ``` [20:51:51 root@centos8 ~]#nload ``` ``` ``` ``` #在nload后面指网卡，可以指定多个，按左右键分别显示网卡状态 ``` ``` [20:52:51 root@centos8 ~]#nload eth0 eth1 ``` ``` ``` ``` #设置刷新闻隔：认刷新闻隔是100毫秒，可通过 -t 命令设置刷新时间（单位是毫秒） ``` ``` [20:54:19 root@centos8 ~]#nload -t 500 eth0 ``` ``` ``` ``` #设置单位：显示种单位一种是显示Bit/s、一种是显示Byte/s，默认是以Bit/s，也可不显示/s ``` ``` #-u h|b|k|m|g|H|B|M|G 表示的含义： h: auto, b: Bit/s, k: kBit/s, m: MBit/s, H: auto, B: Byte/s, K: kByte/s, M: MByte ``` 原文链接：[进程，系统性能和计划任务 2](#)

s

```
</span></span><span class="highlight-line"><span class="highlight-cl">[20:54:52 root@centos8 ~]#nload -u M eth0
</span></span></code></pre>
<h2 id="2-16-网络监视工具iptraf-ng">2.16 网络监视工具 iptraf-ng</h2>
<p>来自于 iptraf-ng 包,可以进网络进行监控,对终端窗口大小有要求.图形化操作。<br>

/p>
<h2 id="2-17-系统资源统计dstat">2.17 系统资源统计 dstat</h2>
<p>dstat 由 pcsp-system-tools 包提供, 但安装 dstat 包即可, 可用于代替 vmstat,iostat 功能格</p>
<p><strong>格式:</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">dstat [-afv] [options..] [delay [count]]
</span></span></code></pre>
<p><strong>常用选项:</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">-c 显示cpu相关信息
</span></span><span class="highlight-line"><span class="highlight-cl">-C #,#,...,total
</span></span><span class="highlight-line"><span class="highlight-cl">-d 显示disk相关信息
</span></span><span class="highlight-line"><span class="highlight-cl">-D total,sda,sdb,...
</span></span><span class="highlight-line"><span class="highlight-cl">-g 显示page相关统计数据
</span></span><span class="highlight-line"><span class="highlight-cl">-m 显示memory关统计数据
</span></span><span class="highlight-line"><span class="highlight-cl">-n 显示network关统计数据
</span></span><span class="highlight-line"><span class="highlight-cl">-p 显示process关统计数据
</span></span><span class="highlight-line"><span class="highlight-cl">-r 显示io请求相关的统计数据
</span></span><span class="highlight-line"><span class="highlight-cl">-s 显示swapped关的统计数据
</span></span><span class="highlight-line"><span class="highlight-cl">--tcp
</span></span><span class="highlight-line"><span class="highlight-cl">--udp
</span></span><span class="highlight-line"><span class="highlight-cl">--unix
</span></span><span class="highlight-line"><span class="highlight-cl">--raw
</span></span><span class="highlight-line"><span class="highlight-cl">--socket
</span></span><span class="highlight-line"><span class="highlight-cl">--ipc
</span></span><span class="highlight-line"><span class="highlight-cl">--top-cpu: 显示占用CPU的进程
</span></span><span class="highlight-line"><span class="highlight-cl">--top-io: 显示最用io的进程
</span></span><span class="highlight-line"><span class="highlight-cl">--top-mem: 显示占用内存的进程
</span></span><span class="highlight-line"><span class="highlight-cl">--top-latency: 显延迟最大的进程
</span></span></code></pre>
<p><strong>范例:</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[20:58:56 root@centos8 ~]#dstat 1 3
</span></span></code></pre>
```

 <https://b3logfile.com/file/2021/01/clipboard-4d02ce62.png?imageView2/2/interlace/1/format/jpg>


2.18 综合监控工具 glances

此工具可以通过 EPEL 源安装,CentOS 8 目前没有提供(已提供,但测试问题)

格式:

```
glances [-bdehmnrsvyz1] [-B bind] [-c server] [-C conffile] [-p port] [-P password] [--password] [-t refresh] [-f file] [-o output]
```

内建命令:

 <https://b3logfile.com/file/2021/01/clipboard-eada862d.png?imageView2/2/interlace/1/format/jpg>

常用选项:

```
-b: 以Byte为单位显示网卡数据速率  
-d: 关闭磁盘I/O模
```

```
-f /path/to/somefile: 设定输入文件位置
```

```
-o {HTML|CSV}: 出格式
```

```
-m: 禁用mount模
```

```
-n: 禁用网络模块
```

```
-t #: 延迟时间间隔
```

```
-1: 每个CPU的相数据单独显示
```

C/S 模式下运行 glances 命令

服务器模式:

glances -s -B IPADDR

IPADDR: 指明监听的本机哪个地址,端口默认为 61209/tcp

客户端模式:

glances -c IPADDR

IPADDR: 要连入的服务器端地址注意: 不同版本不兼容

注意: 不同版本不兼容

2.19 查看进程打开文件 lsof

lsof: list open files, 查看当前系统文件的工具。在 linux 环境下, 一切皆文件, 用户通过文

不仅可以访问常规数据，还可以访问网络连接和硬件如传输控制协议 (TCP) 和用户数据报协议 (UDP) 接字等，系统在后台都为该应用程序分配了一个文件描述符

<p>命令选项:</p>

<pre><code class="highlight-chroma">-a: 列出打开文件存在的进程

-c<进程名>列出指定进程所打开的文件

-g: 列出GID号进详情

-d<文件号>列出占用该文件号的进程

+d<目录>;出目录下被打开的文件

+D<目录>递归列出目录下被打开的文件

-n<目录>;出使用NFS的文件

-i<条件>;出符合条件的进程(4、6、协议、:端口、@ip)

-p<进程号>列出指定进程号所打开的文件

-u: 列出UID号进详情

-h: 显示帮助信息

-v: 显示版本信息

-n: 不反向解析网名字

</code></pre>

<p>范例:</p>

<pre><code class="highlight-chroma">#lsdf 列出当前所有打开的文件

[20:59:14 root@c ntos8 ~]#lsdf | head

COMMAND PID TID TASKCMD USER FD TYPE DEVICE SIZE/OFF NODE NAME

systemd 1 root cwd DIR 253,0 244 128 /

systemd 1 root rtd DIR 253,0 244 128 /

systemd 1 root txt REG 253,0 1609264 50745933 /usr/lib/systemd/systemd

systemd 1 root mem REG 253,0 2191808 16963 /usr/lib64/libm-2.28.so

systemd 1 root mem REG 253,0 628744 537890 /usr/lib64/libudev.so.1.6.11

systemd 1 root mem REG 253,0 969832 20530 /usr/lib64/libsepol.so.1

systemd 1 root mem REG 253,0 1805368 76817 /usr/lib64/libunistring.so.2.1.0

systemd 1 root mem REG 253,0 355456 93758 /usr/lib64/libpcap.so.1.9.0

systemd 1 root mem REG 253,0 145984 39813 /usr/lib64/libgpg-error.so.0.24.2


```
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看当前那个进
正在使用此文件
</span></span><span class="highlight-line"><span class="highlight-cl">[21:05:01 root@c
ntos8 ~]#ls -l /var/log/messages
</span></span><span class="highlight-line"><span class="highlight-cl">COMMAND PID
SER FD TYPE DEVICE SIZE/OFF NODE NAME
</span></span><span class="highlight-line"><span class="highlight-cl">rsyslogd 963 root
5w REG 253,0 474858 17000254 /var/log/messages
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看由登录用户
动而非系统启动的进程
</span></span><span class="highlight-line"><span class="highlight-cl">[21:05:17 root@c
ntos8 ~]#ls -l /dev/pts/0
</span></span><span class="highlight-line"><span class="highlight-cl">COMMAND PID
SER FD TYPE DEVICE SIZE/OFF NODE NAME
</span></span><span class="highlight-line"><span class="highlight-cl">bash 1363 root
0u CHR 136,0 0t0 3 /dev/pts/0
</span></span><span class="highlight-line"><span class="highlight-cl">bash 1363 root
1u CHR 136,0 0t0 3 /dev/pts/0
</span></span><span class="highlight-line"><span class="highlight-cl">bash 1363 root
2u CHR 136,0 0t0 3 /dev/pts/0
</span></span><span class="highlight-line"><span class="highlight-cl">bash 1363 root
255u CHR 136,0 0t0 3 /dev/pts/0
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#指定进程号，可
查看该进程打开的文件
</span></span><span class="highlight-line"><span class="highlight-cl">[21:08:15 root@c
ntos8 ~]#ls -l -p 3967
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看程序打开的
件
</span></span><span class="highlight-line"><span class="highlight-cl">[21:10:06 root@c
ntos8 ~]#ls -l -c bc
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看指定用户打
的文件
</span></span><span class="highlight-line"><span class="highlight-cl">[21:11:07 root@c
ntos8 ~]#ls -l -u zhang | more
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看指定目录下
打开的文件，参数+D为递归列出目录下被打开的文件，参数+d为列出目录下被打开的文件
</span></span><span class="highlight-line"><span class="highlight-cl">[21:11:16 root@c
ntos8 ~]#ls -l +D /var/log/
</span></span><span class="highlight-line"><span class="highlight-cl">[21:11:50 root@c
ntos8 ~]#ls -l +d /var/log/
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看所有网络连
，通过参数-i查看网络连接的情况，包括连接的ip、端口等以及一些服务的连接情况，例 如：sshd
。也可以通过指定ip查看该ip的网络连接情况
</span></span><span class="highlight-line"><span class="highlight-cl">[21:12:02 root@c
ntos8 ~]#ls -l -i -n
</span></span><span class="highlight-line"><span class="highlight-cl">[21:12:38 root@c
ntos8 ~]#ls -l -i@127.0.0.1
```

```

</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看端口连接情
, 通过参数-i:端口可以查看端口的占用情况, -i参数还有查看协议, ip的连接情况等
</span></span><span class="highlight-line"><span class="highlight-cl">[21:12:48 root@c
ntos8 ~]#lsof -i :80 -n
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看指定进程打
的网络连接, 参数-i、-a、-p等, -i查看网络连接情况, -a查看存在的进程, -p指定进 程
</span></span><span class="highlight-line"><span class="highlight-cl">[21:13:28 root@c
ntos8 ~]#lsof -i -n -a -p 7531
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#查看指定状态的
络连接, -n:no host names, -P:no port names,-i TCP指定协议, -s指定协议状态通过多个参数可以
晰的查看网络连接情况、协议连接情况等
</span></span><span class="highlight-line"><span class="highlight-cl">[21:14:05 root@c
ntos8 ~]#lsof -n -P -i TCP -s TCP:ESTABLISHED
</span></span></code></pre>
<p><strong>范例: 利用 lsof 恢复正在使用中的误删除的文件</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">[21:16:17 root@centos8 ~]#lsof | grep /home/zhang/fstab
</span></span><span class="highlight-line"><span class="highlight-cl">[21:16:55 root@c
ntos8 ~]#cat /proc/7812/fd/3
</span></span><span class="highlight-line"><span class="highlight-cl">[21:17:02 root@c
ntos8 ~]#cat /proc/7812/fd/3 &gt; /home/zhang/fstab
</span></span></code></pre>
<h2 id="2-20-综合管理平台-webmin">2.20 综合管理平台 webmin</h2>
<p></p>
<p>官网:<a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.webmin.com%2F"
arget="_blank" rel="nofollow ugc">http://www.webmin.com/</a></p>
<p>下载:<a href="https://ld246.com/forward?goto=http%3A%2F%2Fwww.webmin.com%2F
ownload.html" target="_blank" rel="nofollow ugc">http://www.webmin.com/download.html
/a></p>
<p>Webmin 是目前功能最强大的基于 Web 的 Unix 系统管理工具。管理员通过浏览器访问 Webmi
的各种管 理功能并完成相应的管理动作。目前 Webmin 支持绝大多数的 Unix 系统, 这些系统除
各种版本的 linux 以外还包括: AIX、HPUX、Solaris、Unixware、Irix 和 FreeBSD 等</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">[09:12:59 root@centos8 ~]#yum install /root/webmin-1.962-1.noarch.rpm
</span></span><span class="highlight-line"><span class="highlight-cl">[09:13:20 root@c
ntos8 ~]#chkconfig --list
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">Note: This output
hows SysV services only and does not include native
</span></span><span class="highlight-line"><span class="highlight-cl">systemd services.
ysV configuration data might be overridden by native
</span></span><span class="highlight-line"><span class="highlight-cl">systemd configura
ion.
</span></span></code></pre>
<p>If you want to list systemd services use 'systemctl list-unit-files'.<br>
To see services enabled on particular target use<br>
'systemctl list-dependencies [target]'.</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">webmin          0:off  1:off  2:on   3:on   4:off  5:on   6:off

```

```

</span></span><span class="highlight-line"><span class="highlight-cl">[09:15:21 root@c
ntos8 ~]#service webmin start
</span></span><span class="highlight-line"><span class="highlight-cl">[09:15:39 root@c
ntos8 ~]#ss -ntl
</span></span><span class="highlight-line"><span class="highlight-cl">State      Recv-Q
  Send-Q      Local Address:Port      Peer Address:Port
</span></span><span class="highlight-line"><span class="highlight-cl">LISTEN      0
128          0.0.0.0:111             0.0.0.0:*
</span></span><span class="highlight-line"><span class="highlight-cl">LISTEN      0
128          0.0.0.0:10000          0.0.0.0:*
</span></span><span class="highlight-line"><span class="highlight-cl">LISTEN      0
128          0.0.0.0:22             0.0.0.0:*
</span></span><span class="highlight-line"><span class="highlight-cl">LISTEN      0
128          [::]:111               [::]:*
</span></span><span class="highlight-line"><span class="highlight-cl">LISTEN      0
128          [::]:22                [::]:*
</span></span></code></pre>
<p></p>
<p>2.21 Centos8 新特性 cockpit</p>
<p>由 cockpit 包提供</p>
<p>Cockpit 是 CentOS 8 取入的新特性，是一个基于 Web 界面的应用，它提供了对系统的图形化
理</p>
<ul>
<li>监控系统活动（CPU、内存、磁盘 IO 和网络流量）</li>
<li>查看系统日志条目</li>
<li>查看磁盘分区的容量</li>
<li>查看网络活动（发送和接收） 查看用户帐户</li>
<li>检查系统服务的状态提取已安装应用的信息</li>
<li>查看和安装可用更新（如果以 root 身份登录）并在需要时重新启动系统</li>
<li>打开并使用终端窗口</li>
</ul>
<p>范例：安装 cockpit</p>
<p>[09:24:24 <a href="https://ld246.com/forward?goto=mailto%3Aroot%40centos8" target
"_blank" rel="nofollow ugc">root@</a><a href="https://ld246.com/member/centos8" aria-
ame="centos8" class="tooltipped__user" target="_blank">centos8</a> ~]#dnf install cockpi
<br>
[09:25:04 <a href="https://ld246.com/forward?goto=mailto%3Aroot%40centos8" target="_bl
nk" rel="nofollow ugc">root@</a><a href="https://ld246.com/member/centos8" aria-name
"centos8" class="tooltipped__user" target="_blank">centos8</a> ~]#systemctl start cockpit
<br>
[09:25:20 <a href="https://ld246.com/forward?goto=mailto%3Aroot%40centos8" target="_bl
nk" rel="nofollow ugc">root@</a><a href="https://ld246.com/member/centos8" aria-name
"centos8" class="tooltipped__user" target="_blank">centos8</a> ~]#ss -ntl<br>
State      Recv-Q      Send-Q      Local Address:Port      Peer Address:Port<br>
LISTEN      0            128         0.0.0.0:111             0.0.0.0:*<br>
LISTEN      0            128         0.0.0.0:22             0.0.0.0:*<br>
LISTEN      0            128         <em>:::9090</em>          <em>:::</em><br>
LISTEN      0            128         [::]:111               [::]:</em><br>
LISTEN      0            128         [::]:22                [::]:*</p>
<h2 id="2-22-信号发送kill">2.22 信号发送 kill</h2>
<p>kill：内部命令，可用来向进程发送控制信号，以实现对进程管理,每个信号对应一个数字，信号
称以 SIG 开头（可省略），不区分大小写</p>

```

<p>显示当前系统可用信号: </p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[09:25:23 root@centos8 ~]#kill -l
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[09:27:54 root@centos8 ~]#trap -l
```

```
</span></span></code></pre>
```

<p>查看帮助: man 7 signal</p>

<p>常用信号: </p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">1) SIGHUP 无须关闭进程而让其重读配置文件
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">2) SIGINT 中止在运行的进程; 相当于Ctrl+c
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">3) SIGQUIT 相当ctrl+\
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">4) SIGKILL 强制死正在运行的进程
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">5) SIGTERM 终止在运行的进程, 默认信号
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">6) SIGCONT 继续行
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">7) SIGSTOP 后台眠
```

```
</span></span></code></pre>
```

<p>指定信号的方法: </p>

信号的数字标识: 1, 2, 9

信号完整名称: SIGHUP, sighup

信号的简写名称: HUP, hup

<p>向进程发送信号: </p>

<p>按 PID: </p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill -l [sigspec]
```

```
</span></span></code></pre>
```

<p>范例: </p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[09:35:10 root@centos8 ~]#kill -1 9548
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[09:37:14 root@centos8 ~]#kill -n 9 11440
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[09:41:30 root@centos8 ~]#kill -s SIGINT 11440
```

```
</span></span></code></pre>
```

<p>按名称: killall 来自于 psmisc 包</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">killall [-SIGNAL] comm...
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[09:45:12 root@centos8 ~]#killall -n 1 vim
```

```
</span></span></code></pre>
```

<p>按模式: </p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">pkill [options] pattern
```

```
</span></span></code></pre>
```

<p>常用选项</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">
```


cl">-SIGNAL

-u uid: effective user, 生效者

-U uid: real user 真正发起运行命令者

-t terminal: 与指终端相关的进程

-l: 显示进程名 (pgrep可用)

-a: 显示完整格式进程名 (pgrep可用)

-P pid: 显示指定程的子进程

</code></pre>

<p>范例：查看 HUP 信号</p>

<pre><code class="highlight-chroma">#许多服务的支持的reload操作，实际就是发送了HUP信号#service httpd reload 即相当于 killall -1 httpd

[root@centos6 ~] grep -A 10 -w reload -m 1 /etc/init.d/httpd

reload() {

echo -n \$"Reloading \$prog: "

if ! LANG=\$HTTP _LANG \$httpd \$OPTIONS -t && /dev/null; then

RETVAL=6

echo \$"not reloading due to configuration syntax error"

failure \$"not reloading \$httpd due to configuration syntax error"

else

Force LSB behaviour from killproc

LSB=1 killproc -p {pidfile} \$httpd -HUP RETVAL=\$?

if [\$RETVAL -eq 7 ; then

[root@centos6 ~]

</code></pre>

<p>范例：利用 0 信号实现进程的健康性检查</p>

<pre><code class="highlight-chroma">[09:45:13 root@centos8 ~]#killall -0 ping

[09:48:48 root@centos8 ~]#echo \$?

0

[09:48:53 root@centos8 ~]#killall -0 ping

ping: no process found

[09:49:02 root@centos8 ~]#echo \$?

1

#此方式有局限性

即使进程处于停止或僵尸状态，此方式仍然认为是进程是健康的

```
</span></span></code></pre>
```

<p>范例: pkill 和 pgrep 支持正则表达式</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[09:51:14 root@centos8 ~]#pkill '^p'
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[09:51:47 root@c  
ntos8 ~]#pgrep -a '^p'
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">11993 ping 127.1  
</span></span></code></pre>
```

2.23 作业管理

<p>Linux 的作业控制</p>

- 前台作业：通过终端启动，且启动后一直占据终端
- 后台作业：可通过终端启动，但启动后即转入后台运行（释放终端）

<p></p>

<p>让作业运行于后台</p>

- 运行中的作业： Ctrl+z
- 尚未启动的作业： COMMAND &

<p>后台作业虽然被送往后台运行，但其依然与终端相关；退出终端，将关闭后台作业。如果希望送后台 后，剥离与终端的关系</p>

- nohup COMMAND &>/dev/null &
- screen; COMMAND tmux;
- COMMAND

<p>查看当前终端所有作业：</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[09:54:35 root@centos8 ~]#jobs
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[1]+ Stopped  
ping 127.1
```

```
</span></span></code></pre>
```

<p>作业控制：</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">fg [[%]JOB_NUM]: 把指定的后台作业调回前台
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">bg [[%]JOB_NUM  
: 让送往后台的作业在后台继续运行
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">kill [%JOB_NUM]  
终止指定的作业
```

```
</span></span></code></pre>
```

<p>范例：后台运行的进程和终端关系</p>

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">#终端1运行后台进程
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[10:03:30 root@c  
ntos8 ~]#ping 127.1 &amp;
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">#终端2 可以查看  
进程
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">[10:02:59 root@c  
ntos8 ~]#ps aux | grep ping
```

```

</span></span><span class="highlight-line"><span class="highlight-cl">root      12100 0.
0.2 32424 2248 pts/0   T   10:03  0:00 ping 127.1
</span></span><span class="highlight-line"><span class="highlight-cl">root      12105 0.
0.1 12112 1092 pts/1   R+   10:04  0:00 grep --color=auto ping
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">#关闭终端1后,在
端2查看不到进程
</span></span><span class="highlight-line"><span class="highlight-cl">[10:04:04 root@c
entos8 ~]#ps aux | grep ping
</span></span><span class="highlight-line"><span class="highlight-cl">root      12108 0.
0.1 12112 1096 pts/1   R+   10:04  0:00 grep --color=auto ping
</span></span></code></pre>
<p><strong>范例: nohup</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[10:05:49 root@centos8 ~]#nohup ping 127.1
</span></span><span class="highlight-line"><span class="highlight-cl">nohup: ignoring i
put and appending output to 'nohup.out'
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">[10:06:42 root@c
entos8 ~]#tail -f nohup.out
</span></span><span class="highlight-line"><span class="highlight-cl">64 bytes from 127
0.0.1: icmp_seq=48 ttl=64 time=0.031 ms
</span></span><span class="highlight-line"><span class="highlight-cl">64 bytes from 127
0.0.1: icmp_seq=49 ttl=64 time=0.040 ms
</span></span><span class="highlight-line"><span class="highlight-cl">64 bytes from 127
0.0.1: icmp_seq=50 ttl=64 time=0.024 ms
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">[10:07:15 root@c
entos8 ~]#nohup ping 127.0.0.1 &&/dev/null &&
</span></span><span class="highlight-line"><span class="highlight-cl">[1] 12181
</span></span><span class="highlight-line"><span class="highlight-cl">[10:07:37 root@c
entos8 ~]#pstree -p | grep ping
</span></span><span class="highlight-line"><span class="highlight-cl">├─sshd(951)─┬─s
hd(12112)─┬─sshd(12114)─┬─bash(12115)─┬─ping(12181)
</span></span><span class="highlight-line"><span class="highlight-cl">#关闭对应的终端
观察进程的父进程
</span></span><span class="highlight-line"><span class="highlight-cl">[10:07:13 root@c
entos8 ~]#pstree -p
</span></span><span class="highlight-line"><span class="highlight-cl">├─ping(12181)
</span></span></code></pre>
<h2 id="2-24-并行运行">2.24 并行运行</h2>
<p>利用后台执行, 实现并行功能, 即同时运行多个进程, 提高效率</p>
<p><strong>方法 1</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">cat all.sh
</span></span><span class="highlight-line"><span class="highlight-cl">f1.sh&&
</span></span><span class="highlight-line"><span class="highlight-cl">f2.sh&&
</span></span><span class="highlight-line"><span class="highlight-cl">f3.sh&&
</span></span></code></pre>
<p><strong>方法 2</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl"><span class="highlight-cl">(f1.sh&&);(f2.sh&&);(f3.sh&&);
</span></span></code></pre>
<p><strong>方法 3</strong></p>

```

```
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">f1.sh&amp;f2.sh&amp;f3.sh&amp;
</span></span></code></pre>
<p><strong>范例：多组命令实现并行</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">[root@centos8 ~]#{ ping -c3 127.1; ping 127.2; }&amp; { ping -c3 127.3 ;ping 127.4;}&am
;
</span></span></code></pre>
<p><strong>范例：</strong></p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">NET=192.168.10
</span></span><span class="highlight-line"><span class="highlight-cl">for i in {1..254};do
</span></span><span class="highlight-line"><span class="highlight-cl">{
</span></span><span class="highlight-line"><span class="highlight-cl">ping -c1 -W1 $NE
.$i &amp;&gt;/dev/null &amp;&amp; echo $NET.$i is up || echo $NET.$i is down
</span></span><span class="highlight-line"><span class="highlight-cl">}&amp;
</span></span><span class="highlight-line"><span class="highlight-cl">done
</span></span><span class="highlight-line"><span class="highlight-cl">wait
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">使用并行可以加快
段扫描速度，wait命令的意思是后台运行结束后自动退出
</span></span></code></pre>
```