

## 2.1 进程的相关命令\_物联网 / 嵌入式工程师 - 慕课网

“ 慕课网慕课教程 2.1 进程的相关命令涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

功能

显示当前进程的状态 (Process Status)

语法

ps [options]

常用语法选项

-A : 列出所有的进程

-e : 与 -A 功能类似

-w : 显示加宽可以显示较多的资讯

-au : 显示较详细的信息

-aux : 显示所有包含其他使用者的进程

示例 : ps -aux 显示所有进程的详细信息

```
ben@ubuntu:~$ ps -aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1   0.0  0.3 170852 12828 ?        Ss   Oct03   0:12 /sbin/init auto noprompt
root         2   0.0  0.0      0     0 ?        S    Oct03   0:00 [kthreadd]
root         3   0.0  0.0      0     0 ?        I<   Oct03   0:00 [rcu_gp]
root         4   0.0  0.0      0     0 ?        I<   Oct03   0:00 [rcu_par_gp]
root         5   0.0  0.0      0     0 ?        I<   Oct03   0:00 [netns]
root         7   0.0  0.0      0     0 ?        I<   Oct03   0:00 [kworker/0:0H-events_highpri]
root         9   0.0  0.0      0     0 ?        I<   Oct03   0:13 [kworker/0:1H-events_highpri]
root        10   0.0  0.0      0     0 ?        I<   Oct03   0:00 [mm_percpu_wq]
root        11   0.0  0.0      0     0 ?        S    Oct03   0:00 [rcu_tasks_rude_]
root        12   0.0  0.0      0     0 ?        S    Oct03   0:00 [rcu_tasks_trace]
root        13   0.0  0.0      0     0 ?        S    Oct03   0:02 [ksoftirqd/0]
```

示例 : ps -ef 列出所有的进程，相比 ps -aux 信息要少一些

```
ben@ubuntu:~$ ps -ef
UID        PID  PPID  C  STIME TTY          TIME CMD
root         1    0  0  Oct03 ?        00:00:12 /sbin/init auto noprompt
root         2    0  0  Oct03 ?        00:00:00 [kthreadd]
root         3    2  0  Oct03 ?        00:00:00 [rcu_gp]
root         4    2  0  Oct03 ?        00:00:00 [rcu_par_gp]
root         5    2  0  Oct03 ?        00:00:00 [netns]
root         7    2  0  Oct03 ?        00:00:00 [kworker/0:0H-events_highpri]
root         9    2  0  Oct03 ?        00:00:13 [kworker/0:1H-events_highpri]
root        10    2  0  Oct03 ?        00:00:00 [mm_percpu_wq]
root        11    2  0  Oct03 ?        00:00:00 [rcu_tasks_rude_]
root        12    2  0  Oct03 ?        00:00:00 [rcu_tasks_trace]
```

示例 : ps -ef | grep “可执行文件名” 根据名称查找指定名字

```
ben@ubuntu:~/class/week11$ ps -ef | grep "a.out"
ben          65312   4774  98  14:21 pts/3      00:00:15 ./a.out
```

功能

实时显示进程的信息

语法

top [-] [d delay] [q] [c] [S] [s] [i] [n] [b]

选项

- d : 改变显示的更新速度，或是在交谈式指令列 (interactive command) 按 s
- q : 没有任何延迟的显示速度，如果使用者是有 superuser 的权限，则 top 将会以最高的优先序执行
- c : 切换显示模式，共有两种模式，一是只显示执行档的名称，另一种是显示完整的路径与名称
- S : 累积模式，会将已完成或消失的子进程 (dead child process) 的 CPU time 累积起来
- s : 安全模式，将交谈式指令取消，避免潜在的危机
- i : 不显示任何闲置 (idle) 或无用 (zombie) 的进程
- n : 更新的次数，完成后将会退出 top
- b : 批次档模式，搭配 “n” 参数一起使用，可以用来将 top 的结果输出到档案内

示例 1：显示实时信息

```
ben@ubuntu:~/class/week11$ top

top - 14:30:31 up 7 days, 18:47, 1 user, load average: 1.00, 0.88, 0.48
Tasks: 289 total, 2 running, 286 sleeping, 0 stopped, 1 zombie
%Cpu(s): 50.1 us, 0.0 sy, 0.0 ni, 49.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3889.9 total, 366.6 free, 1534.8 used, 1988.5 buff/cache
MiB Swap: 2048.0 total, 2035.2 free, 12.8 used, 2083.3 avail Mem

  PID USER      PR  NI  VIRT  RES  SHR S %CPU  %MEM    TIME+  COMMAND
 65312 ben        20   0   2364   580   512 R 100.0   0.0   9:13.52 a.out
 65352 ben        20   0  11992  3856  3080 R   0.7   0.1   0:00.02 top
 1686  ben        20   0 4220432 283168 89628 S   0.3   7.1  18:33.50 gnome-shell
    1 root         20   0 170852  12828  8168 S   0.0   0.3   0:12.10 systemd
    2 root         20   0         0         0         0 S   0.0   0.0   0:00.15 kthreadd
    3 root          0 -20         0         0         0 I   0.0   0.0   0:00.00 rcu_gp
    4 root          0 -20         0         0         0 I   0.0   0.0   0:00.00 rcu_par_gp
```

top – 14:34:29 up 7 days, 18:51, 1 user, load average: 1.00, 0.95, 0.61

- top: 名称
- 14:34:29 : 系统当前时间
- up 7 days, 14:30: 系统以及运行的时间，和 uptime 命令相等
- 1 users: 当前有 1 个用户在线
- load average: 1.00, 0.95, 0.61: 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。

Tasks: 290 total, 2 running, 287 sleeping, 0 stopped, 1 zombie

- Tasks: 任务，也就是进程
- 290 total: 当前总共有 290 个任务，也就是 290 个进程
- 2 running: 2 个进程正在运行
- 287 sleeping: 263 个进程正在休眠

- 0 stopped: 0 个停止的进程
- 1 zombie: 1 个僵尸进程

%Cpu(s): 51.0 us, 0.7 sy, 0.0 ni, 47.8 id, 0.0 wa, 0.0 hi, 0.5 si, 0.0 st

- %Cpu(s): CPU 使用率
- 51.0 us: 用户空间占用 CPU 时间的百分比 (大部分进程都运行在用户态, 通常都是希望用户空间 CPU 越高越好)
- 0.7 sy: 内核空间占用 CPU 时间的百分比 (Linux 内核态占用的 CPU 时间, 系统 CPU 占用越高, 表明系统某部分存在瓶颈。通常这个值越低越好)
- 0.0 ni: 占用 CPU 时间的百分比 (ni 是 nice 的缩写, 进程用户态的优先级, 如果调整过优先级, 那么展示的就是调整过 nice 值的进程消耗掉的 CPU 时间, 如果系统中没有进程被调整过 nice 值, 那么 ni 就显示为 0)
- 47.8 id: 空闲 CPU 占用率, 等待进程运行
- 0.0 wa: 等待输入输出的 CPU 时间百分比 (CPU 的处理速度是很快, 磁盘 IO 操作是非常慢的。wa 表示 CPU 在等待 IO 操作完成所花费的时间。系统不应该花费大量的时间来等待 IO 操作, 否则就说明 IO 存在瓶颈)
- 0.0 hi: CPU 硬中断时间百分比 (硬中断是硬盘、网卡等硬件设备发送给 CPU 的中断消息)
- 0.5 si: CPU 软中断时间百分比 (软中断是由程序发出的中断)
- 0.0 st: 被强制等待 (involuntary wait) 虚拟 CPU 的时间, 此时 Hypervisor 在为另一个虚拟处理器服务。

MiB Mem : 3889.9 total, 366.0 free, 1535.2 used, 1988.6 buff/cache

- MiB Mem: 内存
- 3889.9 total: 物理内存总量
- 366.0 free: 空闲内存量
- 1535.2 used: 已使用的内存量
- 1988.6 buff/cache: 用作内核缓存的内存量

MiB Swap: 2048.0 total, 2035.2 free, 12.8 used. 2082.9 avail Mem

- MiB Swap: 交换空间 (虚拟内存, 当内存不足的时候, 把一部分硬盘空间虚拟成内存使用)
- 2048.0 total: 交换区总量
- 2035.2 free: 空闲交换区总量
- 12.8 used: 使用的交换区总量
- 2082.9 avail Mem: 可用于启动一个新应用的内存 (物理内存), 和 free 不同, 它计算的是可回收的 page cache 和 memory slab

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND

- PID: 进程 id
- USER: 进程所有者
- PR: 进程的优先级, 越小优先级越高

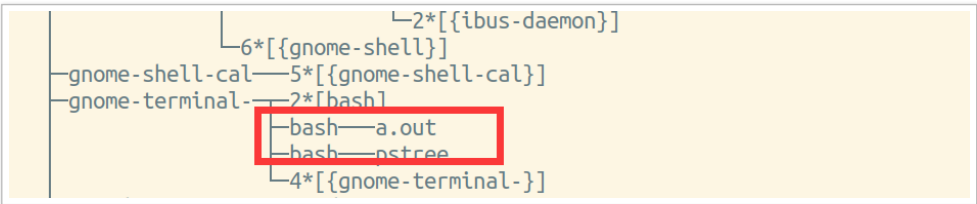
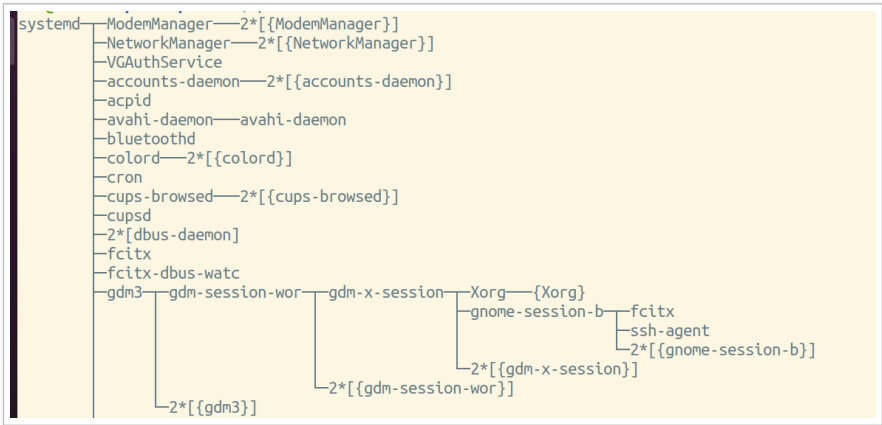
- NI: nice 值, 负值表示高优先级, 正值表示低优先级
- VIRT: 进程使用的虚拟内存, 单位是 kb
- RES: 进程使用的物理内存, 单位 kb
- SHR: 进程使用的共享内存, 单位 kb
- S: 进程状态 (S 表示休眠, R 表示正在运行, Z 表示僵死状态, N 表示该进程优先值为负数, I 表示空闲状态)

示例 2 : top -p <进程 id>

```
top - 15:03:42 up 7 days, 19:20, 1 user, load average: 1.03, 1.03, 1.00
Tasks: 1 total, 0 running, 1 sleeping, 0 stopped, 0 zombie
%Cpu(s): 50.0 us, 0.0 sy, 0.0 ni, 46.7 id, 0.0 wa, 0.0 hi, 3.3 si, 0.0 st
MiB Mem : 3889.9 total, 364.4 free, 1536.6 used, 1988.9 buff/cache
MiB Swap: 2048.0 total, 2035.2 free, 12.8 used. 2081.5 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM    TIME+  COMMAND
 14640 ben        20   0  11140   5688  3720  S   0.0   0.1   0:01.59 bash
```

- pstree 命令是将所有的进程以树型结构的方式进行展示



功能

kill 命令是用于结束进程的命令或者用于显示相关信号

语法

kill [选项] [参数]

选项

示例 : 结束 a.out 进程

- 具体操作步骤如下:
  - step 1: 查询 a.out 进程号
  - |     |       |      |    |       |       |          |         |
|-----|-------|------|----|-------|-------|----------|---------|
| ben | 65312 | 4774 | 99 | 14:21 | pts/3 | 00:54:00 | ./a.out |
|-----|-------|------|----|-------|-------|----------|---------|
  - step 2 : 使用 kill -9 命令结束进程
    - 程序正在运行

- ```
ben@ubuntu:~/class/week12$ ./a.out
```
- 使用 kill 命令之后, 进程结束
- ```
ben@ubuntu:~/class/week12$ ./a.out
Killed
ben@ubuntu:~/class/week12$
```

全文完

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎 beta，[点击查看详细说明](#)

